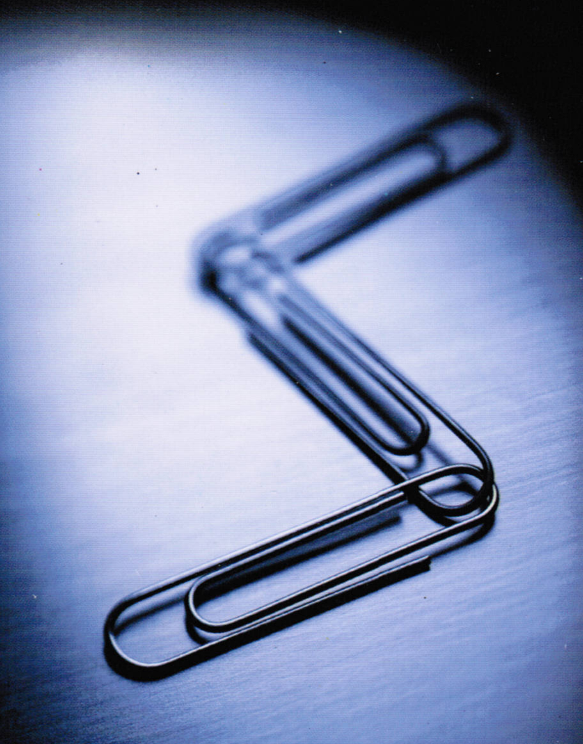


مركز البحوث

تصميم وتطبيق نظم قواعد البيانات العلاقية



تأليف

د. يوسف بن جاسم الهميلي

بسم الله الرحمن الرحيم



مركز البحوث

تصميم وتطبيق نظم قواعد البيانات العلاقية

تأليف

د. يوسف بن جاسم الهميلي

١٤٢٩هـ - ٢٠٠٨م

بطاقة الفهرسة

ⓗ معهد الإدارة العامة، ١٤٢٩هـ.

فهرسة مكتبة الملك فهد الوطنية أثناء النشر.

الهميلي، يوسف بن جاسم

تصميم وتطبيق نظم قواعد البيانات العلاقية. / يوسف بن جاسم الهميلي

الرياض، ١٤٢٩هـ

٥٢٠ ص؛ ١٧ × ٢٤ سم.

ردمك: ١-١٧٠-١٤-٩٩٦٠-٩٧٨

١- الحواسيب - تصميم ٢- قواعد المعلومات
٣- معالجة البيانات أ- العنوان

ديوى ٠٠٥,٧٤ ١٤٢٩/٢٠٣٩

رقم الإيداع: ١٤٢٩/٢٠٣٩
ردمك: ١-١٧٠-١٤-٩٩٦٠-٩٧٨

المحتويات

الموضوع	الصفحة
مقدمة	١٩
الفصل الأول: تطوير نظم المعلومات	٢٥
١-١ ميــزات النظم التطبيقية المبنية على قواعد البيانات	٢٦
١-١-١ تطوير النظم باستخدام الملفات التقليدية	٢٦
١-١-٢ تطوير النظم باستخدام قواعد البيانات	٣١
١-٢ نظم قواعد البيانات	٣٣
١-٢-١ تطوير نظم المعلومات	٣٤
١-٢-١-١ التخطيط الإستراتيجى لنظم المعلومات	٣٦
١-٢-١-٢ دورة حياة تطوير النظم المعلوماتية	٣٩
١-٢-١-٢-١ عمليــة تطوير قاعدة البيانات	٤١
١-٢-١-٣ طرق التطوير البديلة لنظم المعلومات	٤٣
١-٢-١-٣-١ النمــوذج الأولــي (Prototyping)	٤٤
١-٢-٢ مكونات بيئة نظام قواعد البيانات	٤٦
١-٢-٣ مستويات التجريد: المنظورات الثلاثة	٤٩
١-٢-٤ أنواع قواعد البيانات المتوافرة على المســتوى التجارى	٥٢
١-٣ سرد تاريخى لتطور نظم قواعد البيانات	٥٢
الفصل الثانى: نمذجة بيانات المنظمة	٥٧
٢-١ نمذجة البيانات وقواعد العمل باستخدام النمذجة المفاهيمية	٥٨
٢-١-١ مكونات نموذج البيانات كينونة - علاقة	٥٩
٢-١-٢ المكونات الأساسية لنموذج البيانات كينونة - علاقة	٦٣
٢-١-٢-١ الكينونــة (Entity)	٦٣
٢-١-٢-١-١ الفــرق بين فئة الكينونة وحالة من حالات الكينونة	٦٣
٢-١-٢-١-٢ خصائص الكينونات	٦٥
٢-١-٢-١-٣ الخاصية المميزة لفئة الكينونة	٦٩
٢-١-٢-١-٤ قواعد تسمية الخصائص	٧٢
٢-١-٢-١-٥ الكينونة الضعيفة	٧٢
٢-١-٢-٢ العلاقــات (Relationships)	٧٥

الموضوع	الصفحة

٢-١-٢-٢-١ خصائص العلاقة ٧٧
٢-١-٢-٢-٢ الكينونة المشاركة ٧٨
٢-١-٢-٢-٣ درجة العلاقة ٨٠
٢-١-٢-٢-٤ قيود التعددية ٨٢
٢-١-٢-٣ حالة تطبيقية ٨٦
الفصل الثالث: نموذج كينونة - علاقة المطور ٩٥
٣-١ الأنواع الرئيسية والأنواع الفرعية فى نموذج كينونة - علاقة المطور ... ٩٥
٣-١-١ المفاهيم الأساسية والرموز المستخدمة فى الأنواع الرئيسية والأنواع الفرعية ٩٦
٣-١-١-١ توريث الخصائص والعلاقات ٩٩
٣-١-١-٢ توصيف القيود فى علاقات الأنواع الرئيسية والأنواع الفرعية ١٠٢
٣-١-١-٢-١ قيد التخصيص ١٠٣
٣-١-١-٢-١-١ التخصيص الكامل ١٠٣
٣-١-١-٢-١-٢ التخصيص الجزئى ١٠٤
٣-١-١-٢-٢ قيد الانفصال ١٠٥
٣-١-١-٢-٢-١ الانفصال الكامل ١٠٦
٣-١-١-٢-٢-٢ الانفصال المتداخل ١٠٧
٣-١-١-٣ تعريف مميز للأنواع الفرعية ١٠٩
٣-١-١-٣-١ الانفصال الكامل ١٠٩
٣-١-١-٣-٢ الانفصال المتداخل ١١١
٣-١-١-٤ التعميم والتخصيص ١١٢
٣-١-١-٤-١ التعميم ١١٢
٣-١-١-٤-٢ التخصيص ١١٤
٣-١-١-٥ هرميات الأنواع الرئيسية والأنواع الفرعية ١١٧
٣-١-١-٦ التجميع ١٢٠
الفصل الرابع: النموذج العلاقى ولغاته الرسمية ١٢٣
٤-١ نموذج البيانات العلاقى ١٢٣
٤-١-١ المفاهيم الأساسية فى النموذج العلاقى ١٢٤

الموضوع	الصفحة

٤-١-١-١ هيكل البيانات العلاقى ١٢٤
٤-١-١-٢ المفاتيح فى النموذج العلاقى ١٢٦
٤-١-١-٣ خصائص العلاقات (أو الجداول) فى النموذج العلاقى ١٢٩
٤-١-١-٤ قيود التكامل فى النموذج العلاقى ١٣٠
٤-١-١-٤-١ قيود المجال ١٣١
٤-١-١-٤-٢ قيود تكامل الجدول (أو العلاقة) ١٣٢
٤-١-١-٤-٣ قيود القيم غير المعرفة ١٣٣
٤-١-١-٤-٤ قيود السلامة المرجعية ١٣٤
٤-١-١-٤-٥ التعامل مع اختراق القيود أثناء عمليات التعديل على قاعدة البيانات ١٣٦
٤-٢ الجبر العلاقى ١٤٣
٤-٢-١ العمليات الأحادية ١٤٤
٤-٢-١-١ عملية الاختيار ١٤٤
٤-٢-١-٢ عملية الإسقاط ١٤٩
٤-٢-١-٣ عملية إعادة التسمية ١٥٣
٤-٢-٢ العمليات الثنائية ١٥٥
٤-٢-٢-١ عمليات الجبر العلاقى الثنائية من نظرية المجموعات ١٥٥
٤-٢-٢-١-١ عملية الاتحاد ١٥٦
٤-٢-٢-١-٢ عملية التقاطع ١٥٨
٤-٢-٢-١-٣ عملية الفرق ١٦٠
٤-٢-٢-٢ عمليات الجبر العلاقى الثنائية الخاصة بالنموذج العلاقى ١٦٢
٤-٢-٢-٢-١ عملية الضرب الكرتيزى ١٦٢
٤-٢-٢-٢-٢ عملية الربط ١٦٤
٤-٢-٢-٢-٣ عملية القسمة ١٦٦
٤-٣ الحساب العلاقى ١٧٠
٤-٣-١ متغيرات السجلات ١٧١
٤-٣-٢ التعابير والتراكيب فى الحساب العلاقى ١٧٣
٤-٣-٢-١ التعابير الآمنة ١٧٤

الموضوع	الصفحة

٤-٤ أمثلة على استخدام الجبر العلاقى والحساب العلاقى ١٨١

الفصل الخامس: التصميم المنطقى لنظم قواعد البيانات العلاقية ١٨٥

٥-١ التحويل من النمـوذج المفاهيمى كينونة - علاقة إلى النموذج العلاقى ١٨٦

٥-١-١ قاعـدة التحويل الأولى: التعامل مع الكينونات القوية (أو العادية) وخصائصها ١٨٧

٥-١-١-١ التعامل مع الخاصية متعددة القيم ١٨٨

٥-١-٢ قاعدة التحويل الثانية: التعامل مع الكينونات الضعيفة ١٩١

٥-١-٣ قاعـدة التحويل الثالثة: التعامل مع العلاقات الثنائية ١٩٣

٥-١-٣-١ التعامـل مع العلاقات الثنائية ذات التعددية واحد - متعدد ١٩٣

٥-١-٣-٢ التعامـل مع العلاقات الثنائية ذات التعددية متعدد - متعدد ١٩٦

٥-١-٣-٣ التعامـل مع العلاقات الثنائية ذات التعددية واحد - واحد .. ١٩٨

٥-١-٤ قاعدة التحويل الرابعة: التعامل مع الكينونات المشـاركة ٢٠١

٥-١-٤-١ التعامل مع الكينونات المشـاركة عند عدم وجود معرف ٢٠١

٥-١-٤-٢ التعامل مع الكينونات المشاركة عند وجود معرف ٢٠٣

٥-١-٥ قاعدة التحويل الخامسـة: التعامل مع العلاقات الأحادية ٢٠٦

٥-١-٥-١ التعامل مع العلاقات الأحادية ذات التعددية واحد - متعدد ٢٠٦

٥-١-٥-٢ التعامـل مع العلاقات الأحادية ذات التعددية متعدد - متعدد ٢٠٧

٥-١-٦ قاعدة التحويل السادسـة: التعامل مع العلاقات الثلاثية (وما أعلى من ذلك) ٢٠٩

٥-١-٧ قاعدة التحويل السـابعة: التعامل مع علاقات الأنواع الرئيسـية والأنواع الفرعية ٢١١

٥-١-٧-١ الخيار الأول ٢١١

٥-١-٧-٢ الخيار الثانى ٢١٣

٥-١-٧-٣ الخيار الثالث ٢١٥

٥-١-٧-٤ الخيار الرابع ٢١٦

٥-١-٧-٥ فـوارق خيارات تصميم علاقات الأنواع الرئيسـية والأنواع الفرعية ٢١٨

٥-١-٧-٦ تحويل هرميات الأنواع الرئيسية والأنواع الفرعية ٢١٩

الموضوع	الصفحة
٥-١-٨ قاعــدة التحويل الثامنة: التعامل مع التجميع	٢٢١
٥-٢ التصميم المنطقى للحالة الدراسية	٢٢٣
الفصل الســادس: تطبيع العلاقات والتصميم المادى لقواعد البيانات العلاقية	**٢٢٧**
٦-١ التطبيع	٢٢٧
٦-١-١ الجداول جيدة البناء	٢٢٨
٦-١-٢ مستويات التطبيع	٢٣٠
٦-١-٣ الاعتماديات الوظيفية	٢٣١
٦-١-٣-١ الشكل الطبيعى الأول	٢٣٥
٦-١-٣-٢ الشكل الطبيعى الثانى	٢٣٩
٦-١-٣-٣ الشكل الطبيعى الثالث	٢٤٣
٦-١-٣-٤ الشكل الطبيعى بويس - كود	٢٤٧
٦-١-٤ قواعد الاستدلال	٢٥٢
٦-١-٥ خواص التجزئة	٢٥٧
٦-١-٥-١ خاصية المحافظة على الاعتماديات الوظيفية	٢٥٧
٦-١-٥-٢ خاصية السجلات غير الزائفة	٢٥٩
٦-١-٥-٣ التجزئــة التى تتحلى بخاصيــة المحافظة على الاعتماديات الوظيفية وخاصية السجلات غير الزائفة	٢٦١
٦-١-٥-٤ التجزئة إلى الشكل الطبيعى بويس - كود مع المحافظة على خاصيةالسجلاتغيرالزائفة	٢٦٣
٦-١-٦ الشكل الطبيعى الرابع	٢٦٧
٦-١-٧ الأشكال الطبيعية العليا	٢٧٠
٦-٢ التصميم المادى لقواعد البيانات العلاقية	٢٧١
٦-٢-١ عملية التصميم المادى	٢٧١
٦-٢-١-١ تصميم الحقول	٢٧٣
٦-٢-١-٢ تصميم السجلات وعملية فك التطبيع	٢٧٥
٦-٢-١-٣ تنظيم الملفات	٢٨١
٦-٢-١-٤ إنشاء واستخدام الفهارس	٢٨٣
٦-٢-١-٤-١ إنشاء الفهارس ذات المفاتيح الفريدة	٢٨٤

الموضوع	الصفحة

٦-٢-١-٤-٢ إنشاء الفهارس ذات المفاتيح الثانوية (أو غير الفريدة) ٢٨٥

٦-٢-١-٤-٣ استخدامات الفهارس ٢٨٥

الفصل السابع: لغة الاستفسار البنائية - الجزء الأول ٢٨٩

٧-١ لغة تعريف البيانات ٢٩١

٧-١-١ تعليمة الإنشاء ٢٩١

٧-١-١-١ إنشاء قاعدة البيانات ٢٩١

٧-١-١-٢ إنشاء جدول ٢٩٣

٧-١-١-٢-١ أنواع البيانات ٢٩٥

٧-١-١-٣ توصيف القيود فى لغة الاستفسار البنائية والتعامل معها ... ٢٩٧

٧-١-١-٣-١ قيود الحقول والمدى ٢٩٧

٧-١-١-٣-١-١ إنشاء المدى ٢٩٩

٧-١-١-٣-٢ قيود المفاتيح الرئيسية والسلامة المرجعية ٣٠١

٧-١-١-٣-٣ قيود السجلات ٣٠٥

٧-١-١-٣-٤ قيود عامة ٣٠٥

٧-١-١-٣-٥ تعديل القيود والتحكم فى تطبيقها ٣٠٦

٧-١-١-٣-٥-١ تعديل القيود ٣٠٦

٧-١-١-٣-٥-٢ إزالة القيود ٣٠٧

٧-١-١-٣-٥-٣ إضافة القيود ٣٠٨

٧-١-١-٣-٥-٤ تعطيل عمل القيود واستعادة العمل بها ٣٠٨

٧-١-١-٣-٥-٥ تأخير العمل بالقيود ٣٠٩

٧-١-١-٤ إنشاء منظور ٣١١

٧-١-١-٥ إنشاء فهرس ٣١٥

٧-١-٢ تعليمة الإزالة ٣١٧

٧-١-٣ تعليمة التعديل ٣١٨

٧-٢ لغة معالجة البيانات ٣٢٠

٧-٢-١ تعليمة الاختيار ٣٢٠

٧-٢-١-١ اختيار أعمدة محددة من جدول ٣٢١

الموضوع	الصفحة

٧-٢-١-٢ حذف الصفوف المتكررة من نتيجة تعليمة الاختيار باستخدام كلمة (DISTINCT) ٣٢٧
٧-٢-١-٣ الأسماء المستعارة للأعمدة ٣٣١
٧-٢-١-٤ اختيار كافة أعمدة جـدول ٣٣٤
٧-٢-١-٥ الاسترجاع المشروط ٣٣٦
٧-٢-١-٥-١ العوامل العلاقية (LIKE, BETWEEN and IN ٣٤١
٧-٢-١-٥-١-١ العامل العلاقى «مثل» (LIKE) ٣٤١
٧-٢-١-٥-١-٢ العامل العلاقى «بين» (BETWEEN) ٣٤٣
٧-٢-١-٥-١-٣ العامل العلاقى «فى» (IN) ٣٤٦
٧-٢-١-٥-٢ القيم غيـر المعرفة ٣٤٨
٧-٢-١-٥-٢-١ المنطق الثلاثى القيم ٣٤٩
٧-٢-١-٦ ترتيـب نتيجة عملية الاختيار باسـتخدام عبارة (ORDER BY) ٣٥٣
٧-٢-١-٦-١ ترتيب النتائج وفقاً للأرقام النسـبية للأعمدة ٣٥٦
٧-٢-١-٧ القيم المحسوبة ٣٥٨
٧-٢-١-٨ دوال التجميع (أو الأعمدة) ٣٦٠
٧-٢-١-٩ عبارة التجميع (GROUP BY) ٣٦٤
٧-٢-١-١٠ عبارة ترشيح المجموعات الفرعية (HAVING) ٣٦٧
٧-٢-١-١١ استخدام تعليمات المجموعات لدمج نتائج تعليمات اختيار متعددة ٣٦٨
٧-٢-١-١١-١ الاتحاد ٣٦٨
٧-٢-١-١١-٢ التقاطع ٣٧١
٧-٢-١-١١-٣ الفرق ٣٧٢
الفصل الثامن: لغة الاستفسار البنائية – الجزء الثانى ٣٧٥
٨-١ الضرب الكرتيزى وربط الجداول فى تعليمة الاختيار ٣٧٥
٨-٢ الاستفسارات المتداخلة ٣٨٧
٨-٢-١ العوامل العلاقية (IN, ANY, ALL) ٣٨٩
٨-٢-٢ الاستفسارات المتداخلة المتعددة المستويات ٣٩٣
٨-٢-٣ الاستفسارات المتداخلة المرتبطة ٣٩٤
٨-٢-٣-١ العامل العلاقى (EXISTS) ٣٩٧

الموضوع	الصفحة
٨-٣ تعليمات الإضافة، والحذف، والتحديث	٣٩٩
٨-٣-١ تعليمة الإضافة	٤٠٠
٨-٣-٢ تعليمة الحذف	٤٠٢
٨-٣-٣ تعليمة التحديث	٤٠٣
٨-٤ دوال الوقت والتاريخ، ودوال الأرقام، ودوال السلاســل الحرفية، ودوال التحويل	٤٠٥
٨-٤-١ دوال الوقت والتاريخ	٤٠٦
٨-٤-٢ دوال الأرقام	٤٠٨
٨-٤-٣ دوال السلاسل الحرفية	٤٠٩
٨-٤-٤ دوال التحويل	٤١٠
٨-٥ لغة التحكم فى البيانات	٤١٣
٨-٥-١ منح الصلاحيات	٤١٣
٨-٥-١-١ منح الصلاحيات على المنظورات	٤١٧
٨-٥-١-٢ إعطاء الحق فى تخويل الصلاحية	٤١٧
٨-٥-٢ سحب الصلاحيات	٤١٨
الفصل التاسع: موضوعات متقدمة فى نظم قواعد البيانات	**٤٢١**
٩-١ المعامــلات (Transactions)	٤٢٢
٩-١-١ التأكيــد على خصائص المعاملات فى نظم إدارة قواعد البيانات	٤٢٧
٩-١-١-١ نظام التحكم فى التزامن	٤٢٧
٩-١-١-٢ نظام الاستعادة (أو التشافى)	٤٣١
٩-١-٢ الزنادات والإجراءات المتكررة	٤٣٤
٩-١-٢-١ الزنادات	٤٣٥
٩-١-٢-٢ الإجراءات المتكررة	٤٣٥
٩-٢ قواعد البيانات الشيئية	٤٣٦
٩-٢-١ مفاهيم الأشياء الموجهة	٤٣٧
٩-٢-١-١ مفهوم الشىء	٤٣٨
٩-٢-١-١-١ ذاتية الشىء	٤٣٨
٩-٢-١-١-٢ حالة (أو قيمة) الشىء	٤٣٩
٩-٢-١-١-٣ سلوك (أو عمل) الشىء	٤٣٩

الموضوع **الصفحة**

٩-٢-١-٢ الفئة (أو الصنف) ٤٤٠
٩-٢-١-٢-١ أنواع العمليات ٤٤٢
٩-٢-١-٣ مفهوم التغليف ٤٤٣
٩-٢-١-٤ التحميل الزائد ٤٤٤
٩-٢-١-٥ هرميات الأصناف والتوريث ٤٤٥
٩-٣ قواعد البيانات العلاقية - الشيئية ٤٤٧
٩-٣-١ مفاهيم قواعد البيانات العلاقية - الشيئية ٤٤٩
٩-٣-١-١ خصائص قواعد البيانات العلاقية - الشيئية ٤٤٩
٩-٤ قواعد البيانات الموزعة ٤٥٠
٩-٤-١ خيارات توزيع البيانات ٤٥٤
٩-٤-١-١ تكرار البيانات ٤٥٥
٩-٤-١-٢ التقسيم الأفقى ٤٥٦
٩-٤-١-٣ التقسيم الرأسى ٤٥٦
٩-٤-١-٤ الجمع بين خيارات التوزيع ٤٥٧
المراجع ٤٥٩
ملاحق ٤٦١
ملحق رقم (١): حالة دراسية - قاعدة بيانات الجامعة الأهلية ٤٦٣
ملحق رقم (١)-١ قواعد العمل المعمول بها فى الجامعة ٤٦٣
ملحق رقم (١)-٢ النموذج المفاهيمى لقاعدة البيانات ٤٦٥
ملحق رقم (١)-٣ النموذج المنطقى لقاعدة البيانات ٤٦٦
ملحق رقم (١)-٤ جداول قاعدة البيانات حسب بنائها باستخدام نظام إدارة قاعدة بيانات أكسس ٤٦٧
ملحق رقم (١)-٥ العلاقات بين جداول قاعدة البيانات حسب بنائها باستخدام نظام إدارة قاعدة بيانات أكسس ٤٧٥
ملحق رقم (١)-٦ إنشاء قاعدة البيانات باستخدام تعليمات SQL فى بيئة أوراكل SQL*Plus ٤٧٦
ملحق رقم (١)-٧ استعراض لمحتويات جداول قاعدة البيانات بعد إنشائها فى بيئة أوراكل باستخدام تعليمة (SELECT * FROM TableName) ٤٨٥
ملحق رقم (٢): تمارين تطبيقية على لغة الاستفسار البنائية SQL ٤٩٠

الجداول

الجدول	الصفحة
جدول رقم (١-١): أمثلة لعوامل التخطيط الإستراتيجى	٣٦
جدول رقم (١-٢): غرض كل مرحلة من مراحل «دورة حياة تطوير النظم المعلوماتية» ومخرجاتها	٤٠
جدول رقم (١-٣): الفعاليات المصاحبة لمراحل تطوير النظم المعلوماتية وفق طريقة النموذج الأولى	٤٥
جدول رقم (٤-١): أمثلة لتعاريف مدى بعض الحقول	١٣٢
جدول رقم (٦-١): مثال لجدول جيد البناء	٢٢٨
جدول رقم (٦-٢): مثال لجدول سيئ البناء ناتج عن تصميم مفاهيمى سيئ	٢٣٠
جدول رقم (٦-٣): جدول ليس فى الشكل الطبيعى الأول	٢٣٥
جدول رقم (٦-٤): جدول يحتوى على حقول متعددة القيم ومركبة	٢٣٨

الأشكال

الشكل	الصفحة
شكل رقم (١-١): ارتباط البيانات بالبرامج عند بداية ظهور الحاسبات الآلية	٢٦
شكل رقم (١-٢): تخزين البيانات فى ملفات على القرص الصلب واستخدام مبدأ المشاركة	٢٨
شكل رقم (١-٣): هرمية تطوير نظم المعلومات فى المنظمة وفق منهجية هندسة المعلومات	٣٤
شكل رقم (١-٤): خطوات تطوير التطبيقات وفق منهجية «دورة حياة تطوير النظم المعلوماتية»	٣٩
شكل رقم (١-٥): خطوات تطوير النظم المعلوماتية وفق طريقة النموذج الأولى	٤٤
شكل رقم (١-٦): المكونات الرئيسية لبيئة نظام قواعد البيانات	٤٧
شكل رقم (١-٧): مستويات التجريد فى بيئة نظام قواعد البيانات	٥٠
شكل رقم (٢-١): الرموز الأساسية المستخدمة فى نموذج كينونة - علاقة ومعانيها	٦٠
شكل رقم (٢-٢): أمثلة لفئات الكينونات وطريقة تمثيلها فى مخطط كينونة - علاقة	٦٥
شكل رقم (٢-٣): أمثلة توضح طرق تمثيل الأنواع الأربعة من الخصائص فى نموذج كينونة - علاقة	٦٧
شكل رقم (٢-٤): تمثيل القسم الدراسى كفئة كينونة ذات خاصيتين بسيطتين	٦٨
شكل رقم (٢-٥): تمثيل عضو هيئة التدريس كفئة كينونة ذات أربع خصائص بسيطة وخاصية مركبة	٦٩
شكل رقم (٢-٦): التفريق بين تمثيل الخاصية المميزة وبقية خصائص الكينونة فى مخطط كينونة - علاقة	٧١
شكل رقم (٢-٧): مثال لكينونة ضعيفة وارتباطها بالكينونة المالكة	٧٣
شكل رقم (٢-٨): تمثيل المجموعة الدراسية ككينونة ضعيفة وارتباطها بكينونة المادة الدراسية	٧٥
شكل رقم (٢-٩): تمثيل العلاقات فى مخطط كينونة - علاقة	٧٦

الشكل	الصفحة

شكل رقم (٢-١٠): مثال لعلاقة أعضاء هيئة التدريس بالمواد الدراسية المؤهلين لتدريسها	٧٧
شكل رقم (٢-١١): تمثيل خصائص العلاقة فى مخطط كينونة - علاقة	٧٨
شكل رقم (٢-١٢): تمثيل العلاقة المشاركة فى مخطط كينونة - علاقة	٧٩
شكل رقم (٢-١٣): تمثيل العلاقة الأحادية فى مخطط كينونة - علاقة	٨١
شكل رقم (٢-١٤): تمثيل العلاقة الثنائية فى مخطط كينونة - علاقة	٨١
شكل رقم (٢-١٥): تمثيل العلاقة الثلاثية فى مخطط كينونة - علاقة	٨٢
شكل رقم (٢-١٦): الرموز المستخدمة لتمثيل قيود التعددية فى مخطط كينونة - علاقة	٨٢
شكل رقم (٢-١٧): تمثيل تعددية اختيارى واحد فى مخطط كينونة - علاقة	٨٣
شكل رقم (٢-١٨): تمثيل تعددية إجبارى واحد فى مخطط كينونة - علاقة	٨٣
شكل رقم (٢-١٩): تمثيل تعددية اختيارى متعدد فى مخطط كينونة - علاقة	٨٤
شكل رقم (٢-٢٠): تمثيل تعددية إجبارى متعدد فى مخطط كينونة - علاقة	٨٤
شكل رقم (٢-٢١): تمثيل تعددية العلاقة اختيارى واحد وإجبارى متعدد	٨٥
شكل رقم (٢-٢٢): تمثيل تعددية العلاقة إجبارى واحد وإجبارى متعدد	٨٥
شكل رقم (٢-٢٣): تمثيل تعددية العلاقة إجبارى متعدد وإجبارى متعدد	٨٦
شكل رقم (٢-٢٤): تمثيل تعددية العلاقة اختيارى متعدد وإجبارى متعدد	٨٦
شكل رقم (٢-٢٥): تمثيل قاعدة العمل الأولى للجامعة الأهلية فى مخطط كينونة - علاقة	٨٧
شكل رقم (٢-٢٦): تمثيل قاعدة العمل الثانية للجامعة الأهلية فى مخطط كينونة - علاقة	٨٧
شكل رقم (٢-٢٧): تمثيل قاعدة العمل الثالثة للجامعة الأهلية فى مخطط كينونة - علاقة	٨٨
شكل رقم (٢-٢٨): تمثيل قاعدة العمل الرابعة للجامعة الأهلية فى مخطط كينونة - علاقة	٨٨
شكل رقم (٢-٢٩): تمثيل قاعدة العمل الخامسة للجامعة الأهلية فى مخطط كينونة - علاقة	٨٩

الشكل	الصفحة

شكل رقم (٢-٣٠): تمثيل قاعدة العمل السادسة للجامعة الأهلية فى مخطط كينونة - علاقة ٨٩

شكل رقم (٢-٣١): تمثيل قاعدة العمل السابعة للجامعة الأهلية فى مخطط كينونة - علاقة ٩٠

شكل رقم (٢-٣٢): تمثيل قاعدة العمل الثامنة وقاعدة العمل التاسعة للجامعة الأهلية فى مخطط كينونة - علاقة ٩١

شكل رقم (٢-٣٣): تمثيل قاعدة العمل العاشرة للجامعة الأهلية فى مخطط كينونة - علاقة ٩١

شكل رقم (٢-٣٤): تمثيل قاعدة العمل الحادية عشرة والثانية عشرة للجامعة الأهلية فى مخطط كينونة - علاقة ٩٢

شكل رقم (٢-٣٥): تمثيل قاعدة العمل الثالثة عشرة للجامعة الأهلية فى مخطط كينونة -علاقة ٩٢

شكل رقم (٢-٣٦): تمثيل قاعدة العمل الرابعة عشرة للجامعة الأهلية فى مخطط كينونة - علاقة ٩٣

شكل رقم (٢-٣٧): كامل مخطط كينونة - علاقة للجامعة الأهلية وفق قواعد العمل المعطاة ٩٤

شكل رقم (٣-١): الرموز الأكثر شيوعاً فى تمثيل الأنواع الرئيسية والأنواع الفرعية ٩٧

شكل رقم (٣-٢): النوع الرئيسى لكينونة الموظفين وأنواعها الفرعية الثلاثة ٩٩

شكل رقم (٣-٣): توريث العلاقات من النوع الرئيسى لأنواعه الفرعية ١٠٠

شكل رقم (٣-٤): مثال لإيضاح الحالات التى يفضل فيها استخدام علاقات الأنواع الرئيسية والأنواع الفرعية ١٠٢

شكل رقم (٣-٥): تمثيل التخصيص الكامل فى علاقات الأنواع الرئيسية والأنواع الفرعية ١٠٤

شكل رقم (٣-٦): تمثيل التخصيص الجزئى فى علاقات الأنواع الرئيسية والأنواع الفرعية ١٠٥

شكل رقم (٣-٧): تمثيل الانفصال الكامل فى علاقات الأنواع الرئيسية والأنواع الفرعية ١٠٧

الشكل	الصفحة
شكل رقم (٣-٨): تمثيل الانفصال المتداخل فى علاقات الأنواع الرئيسية والأنواع الفرعية	١٠٨
شكل رقم (٣-٩): التوليفات الأربعة المحتملة لقيد التخصيص وقيد الانفصال	١٠٩
شكل رقم (٣-١٠): تعريف مميز الأنواع الفرعية فى حالة الانفصال الكامل	١١٠
شكل رقم (٣-١١): تعريف مميز الأنواع الفرعية فى حالة الانفصال المتداخل	١١١
شكل رقم (٣-١٢): التعرف على الأنواع الرئيسية والأنواع الفرعية من خلال عملية التعميم	١١٣
شكل رقم (٣-١٣): نمذجة الأنواع الرئيسية والأنواع الفرعية بعد التعرف عليها من خلال عملية التعميم	١١٣
شكل رقم (٣-١٤): التعرف على الأنواع الرئيسية والأنواع الفرعية من خلال عملية التخصيص	١١٤
شكل رقم (٣-١٥): نمذجة الأنواع الرئيسية والأنواع الفرعية بعد التعرف عليها من خلال عملية التخصيص	١١٥
شكل رقم (٣-١٦): تحديد العلاقات الخاصة بالأنواع الفرعية فى علاقات الأنواع الرئيسية والأنواع الفرعية	١١٦
شكل رقم (٣-١٧): أحد الأمثلة الشائعة لهرميات الأنواع الرئيسية والأنواع الفرعية	١١٧
شكل رقم (٣-١٨): إضافة خاصية مميز الأنواع الفرعية فى هرميات الأنواع الفرعية والأنواع الرئيسية	١١٩
شكل رقم (٣-١٩): علاقة «الدعم المالى» التى تربط بين فئة كينونة الأقسام وفئة كينونة المشاريع	١٢٠
شكل رقم (٣-٢٠): علاقة «متابعة سير المشروع» عند استخدام مفهوم التجميع	١٢١
شكل رقم (٣-٢١): إيضاح التعددية عند استخدام مفهوم التجميع فى نمذجة العلاقات	١٢٢
شكل رقم (٤-١): جدول «عضو هيئة التدريس» (FACULTY) وفق النموذج العلاقى	١٢٥
شكل رقم (٤-٢): تمثيل المفاتيح الخارجية فى النموذج العلاقى	١٢٥

الشكل	الصفحة
شكل رقم (٤-٣): استخدام المفتاح الخارجي لربط سجلات الجدول نفسه ببعضها	١٣٥
شكل رقم (٥-١): تحويل الكينونات القوية إلى علاقات (أو جداول)	١٨٨
شكل رقم (٥-٢): تحويل الخاصية المتعددة القيم إلى علاقة (أو جدول)	١٨٩
شكل رقم (٥-٣): تحويل الخاصية المركبة المتعددة القيم إلى علاقة (أو جدول)	١٩٠
شكل رقم (٥-٤): تحويل الكينونة الضعيفة إلى علاقة (أو جدول)	١٩٢
شكل رقم (٥-٥): تحويل العلاقة الثنائية ذات التعددية واحد - متعدد إلى النموذج العلاقي	١٩٤
شكل رقم (٥-٦): تحويل العلاقة الثنائية ذات التعددية واحد - متعدد ترتبط بها كينونة ضعيفة إلى النموذج العلاقي	١٩٥
شكل رقم (٥-٧): تحويل العلاقة الثنائية ذات التعددية متعدد - متعدد إلى النموذج العلاقي	١٩٧
شكل رقم (٥-٨): تحويل العلاقة الثنائية ذات التعددية واحد - واحد إلى النموذج العلاقي	١٩٩
شكل رقم (٥-٩): تحويل الكينونة المشاركة عند عدم وجود معرف إلى النموذج العلاقي	٢٠٢
شكل رقم (٥-١٠): تحويل الكينونة المشاركة عند وجود معرف إلى النموذج العلاقي	٢٠٤
شكل رقم (٥-١١): تحويل العلاقة الأحادية ذات التعددية واحد - متعدد إلى النموذج العلاقي	٢٠٧
شكل رقم (٥-١٢): تحويل العلاقة الأحادية ذات التعددية متعدد - متعدد إلى النموذج العلاقي	٢٠٨
شكل رقم (٥-١٣): تحويل العلاقة الثلاثية إلى النموذج العلاقي	٢١٠
شكل رقم (٥-١٤): تحويل علاقة الأنواع الرئيسية والأنواع الفرعية إلى النموذج العلاقي وفق الخيار الأول للتحويل	٢١٢
شكل رقم (٥-١٥): تحويل علاقة الأنواع الرئيسية والأنواع الفرعية إلى النموذج العلاقي وفق الخيار الثاني للتحويل	٢١٤

الشكل	الصفحة

شكل رقم (٥-١٦): تحويل علاقة الأنواع الرئيسية والأنواع الفرعية إلى النموذج العلاقى وفق الخيار الثالث للتحويل	٢١٥
شكل رقم (٥-١٧): تحويل علاقة الأنواع الرئيسية والأنواع الفرعية إلى النموذج العلاقى وفق الخيار الرابع للتحويل	٢١٧
شكل رقم (٥-١٨): تحويل هرميات الأنواع الرئيسية والأنواع الفرعية إلى النموذج العلاقى	٢٢٠
شكل رقم (٥-١٩): تحويل علاقات التجميع إلى النموذج العلاقى	٢٢٢
شكل رقم (٥-٢٠): التصميم المنطقى الكامل لقاعدة بيانات الجامعة الأهلية	٢٢٤
شكل رقم (٦-١): مثال لتصميم مفاهيمى سيئ	٢٢٩
شكل رقم (٦-٢): مستويات تطبيع العلاقات (أو الجداول)	٢٣١
شكل رقم (٦-٣): نتيجة التطبيع للشكل الطبيعى الأول وفق الطريقة الأولى	٢٣٦
شكل رقم (٦-٤): نتيجة التطبيع للشكل الطبيعى الأول وفق الطريقة الثانية	٢٣٧
شكل رقم (٦-٥): نتيجة التطبيع للشكل الطبيعى الأول وفق الطريقة الثالثة	٢٣٧
شكل رقم (٦-٦): تطبيع جدول ذى حقل متعدد القيم ومركب للشكل الطبيعى الأول	٢٣٩
شكل رقم (٦-٧): الشكل العام للاعتمادية الوظيفية التى تخل بشروط الشكل الطبيعى بويس - كود	٢٥٠
شكل رقم (٦-٨): فك التطبيع بين جدولين تربط بينهما علاقة واحد - واحد	٢٧٧
شكل رقم (٦-٩): فك التطبيع عند وجود علاقة متعدد - متعدد (أى كينونة مشاركة) بين جدولين ولا يوجد للعلاقة مفتاح خاص بها	٢٧٨
شكل رقم (٦-١٠): فك التطبيع عند وجود بيانات مرجعية	٢٧٩
شكل رقم (٧-١): اللغات الفرعية (أو الفئات الثلاث من التعليمات) المكونة للغة الاستفسار البنائية	٢٩٠
شكل رقم (٩-١): الشكل العام لتعريف الزنادات	٤٣٥
شكل رقم (٩-٢): الشكل العام لتعريف الإجراءات المتكررة	٤٣٦
شكل رقم (٩-٣): فئة (أو صنف) الطلبة وحالة (أو واقعة) منها	٤٤١
شكل رقم (٩-٤): مثال توضيحى للفئات (أو الأصناف) الفرعية	٤٤٧
شكل رقم (٩-٥): نموذج مبسط لقاعدة بيانات موزعة	٤٥١

مقدمة:

تعـد المعلومات فى وقتنـا الراهن أحد أهم موارد المنظمـات، وذلك على اختلاف مجالات عملها. ونظراً لأن المادة الأساسية التى تستقى وتستنبط منها المعلومات هى البيانـات فقد حدا هذا بالمنظمات المختلفة إلى جمع البيانات، سـواء التاريخية منها أو الحديثـة، وتخزينها فى أماكن تضمن المحافظة عليها من العبث أو التخريب. هذا بالإضافة للعمل الدؤوب على تحديث هذه البيانات؛ لكونها أحد الموارد الرئيسية والمهمة للمنظمـات. ومع تزايد أهمية البيانات فى المنظمات الحديثة ظهرت الحاجة الماسـة لمكننة طرق حفظ واسـترجاع البيانات بالإضافـة لمعالجتها حتى أضحت هذه الطرق جوهـر أى نظـام معلوماتى حديث. كما أضحت النظم التى تقوم بحفظ واسـترجاع ومعالجة البيانات تعرف بنظم إدارة قواعد البيانات. وبات مجال نظم قواعد البيانات واحـداً مـن أخصب التخصصات العلميـة طَرْقاً من قبل الباحثـين وتطبيقاً من قبل المتخصصين فى مجال تطوير النظم المعلوماتية؛ لكون هذا المجال يتعامل مع كافة أنواع المنظمات، ومنها - على سبيل المثال وليس الحصر- تلك التى تعنى بالتعليم، والرعاية الصحية، والمكتبات، والأعمال البنكية وأسواق المال، والتجارة الإلكترونية.

ونتيجة للأهمية الكبيرة لقواعد البيانات والنظم التى تعنى بإدارتها فقد تم تأليف عدد كبير من الكتب العلمية المتخصصة باللغة الإنجليزية فى هذا المجال موجهة لفئات مختلفـة من القراء، ومن ضمنها الكتب الدراسـية التى تـدرس فى المراحل الجامعية المختلفـة. ويتوافر من بين هذه الكتب حالياً عدد لا بأس به من الكتب الجامعية التى تعد مراجع دراسـية متميزة للطلبة المتخصصين فى مجال الحاسب الآلى فى المرحلة الجامعية وفى السـنوات الأولى من مرحلة الدراسات العليا، ومن ضمنها تلك المدرجة ضمن مراجع هذا الكتاب. وقد ألَّف هذه الكتب أساتذة متخصصون فى مجال قواعد البيانـات لهم باع طويل فى البحـث العلمى والتدريس فى هذا المجال على المسـتوى الدولى. كما أن هذه الكتب قد ظهرت فى طبعات مختلفة حظيت بالكثير من التنقيح والتطوير حتى أصبحت بالشكل المتميز التى هى عليه الآن. وعلى الرغم من أن مجال قواعد البيانات أصبح من المجالات التخصصية المعروفة التى لها كيانها الخاص ضمن تخصصات علوم الحاسب الآلى، إلا أنه يوجد نقص كبير فى مكتبتنا العربية من الكتب العلمية المتخصصة التى تتعاطى مع مجال قواعد البيانات بشـكل علمى يتسم بالعمق والشـمولية من حيث العرض والتحليل والنقاش للمواضيع الرئيسية التى تدور حولها مفاهيم وتقنيات قواعد البيانات ونظمها. وقد شجع هذا القصور النسبى على تأليف هذا الكتاب الذى يشـتمل على المواضيع الرئيسـية لقواعد البيانات وبشكل يميل إلى

الجانب التطبيقى ودون تقصير فى عرض الجوانب النظرية التى تستند إليها مفاهيم وتقنيات قواعد البيانات. ولقد صُرف الوقت الكثير والجهد الكبير فى تأليف هذا الكتاب ليكون بادرة تهدف إلى نقل شىء مما يزخر به مجال قواعد البيانات من مفاهيم وتقنيات إلى المكتبة العربية، وعلى أمل أن يشكل خطوة جادة تساعد على إضافة المزيد من الخطوات المستقبلة إلى الأمام وصولاً إلى كتب علمية متخصصة فى هذا المجال تثرى مقتنيات مكتبتنا العربية وتسهم فى إثرائها للفكر العربى المتخصص.

ويتكون هذا الكتاب من تسعة فصول بالإضافة إلى الملحقين. يستعرض **الفصل الأول** الميزات التى تتحلى بها نظم قواعد البيانات فى بناء وتطوير نظم التطبيقات مقارنة بالطريقة التقليدية المبنية على الملفات فى تطوير النظم المعلوماتية، كما يستعرض الخطوات الرئيسية المتبعة فى تحليل، وتصميم، وبناء، وإدارة قواعد البيانات. ولأن قاعدة بيانات أى نظام معلوماتى تمثل جزءاً من النظام المعلوماتى، فإن هذا الفصل يبين أيضاً موقع عملية تطوير قاعدة البيانات ضمن عملية التطوير الكلى للنظام المعلوماتى. ويمكن أن نلخص أهم عمليات تطوير أية قاعدة بيانات فى: النمذجة المفاهيمية لبيانات المنظمة (أو النظام المعلوماتى)، التصميم المنطقى لقاعدة البيانات، التصميم المادى لقاعدة البيانات وتعريفها، وبناء قاعدة البيانات. ويقدم الفصل فى نهايته أهم الأحداث التاريخية فى تطور نظم قواعد البيانات. وبذلك يعتبر هذا الفصل مدخلاً للتعرف على نظم قواعد البيانات وميزاتها، كما يمثل أساساً لتسلسل بقية فصول الكتاب ومدخلاً لما تحتويه من موضوعات.

الفصل الثانى خصص لموضوع نمذجة بيانات المنظمة التى تعرف عادة بما يسمى «النمذجة المفاهيمية» (Conceptual Data Modeling). فالنمذجة المفاهيمية عبارة عن نمذجة لبيانات المنظمة بشكل عالى المستوى قريب من إدراك المستفيدين، غير المتخصصين، للبيانات التى يتعاملون معها والارتباطات فيما بينها. ويشرح هذا الفصل المكونات الأساسية لنموذج «كينونة – علاقة» (Entity-Relationship Model) الذى يعد أكثر النماذج المفاهيمية عالية المستوى شيوعاً.

خصص **الفصل الثالث** لشرح مفاهيم ومكونات إضافية للنموذج المفاهيمى كينونة – علاقة. والسبب وراء ذلك يرجع إلى تعقيد البيانات والعلاقات فيما بينها فى بعض المنظمات الحديثة؛ مما يستدعى ضرورة إثراء نموذج كينونة – علاقة بمفاهيم تمكن من نمذجة البيانات الأكثر تعقيداً. ومن هذه المفاهيم، التى يتطرق إليها الفصل، «الأنواع الرئيسية» و«الأنواع الفرعية» (Supertype/Subtype)، و«التعميم» (Generalization)، و«التخصيص» (Specialization)، و«التجميع» (Aggregation).

أما **الفصل الرابع** فهو مخصص لشـــرح المفاهيم الأساسية للنموذج العلاقى الذى يعـــد أحد المحاور الرئيســـية للكتاب. فالنموذج العلاقى يعد أنجـــح النماذج التمثيلية للبيانات وأكثرها اســـتخداماً فـــى نظم قواعد البيانات المتوافرة حالياً على المســـتوى التجارى هذا بالإضافة إلى كونه الأكثر اســـتخداماً فى المنظمات الحديثة. ومن أبرز أسباب نجاح وانتشار استخدام هذا النموذج سهولته فى تمثيل البيانات بالإضافة إلى اســـتناده إلى أســـس رياضية صلبة تمكنه من التعامل مع البيانات وحســـاب نتائجها. لذلك فإن هذا الفصل يستعرض أيضاً لغتين من لغات النموذج العلاقى الرسمية وهما: الجبر العلاقى (Relational Algebra) - التى تعد إحدى لغات النموذج العلاقى الإجرائية (Procedural Language)؛ والحســـاب العلاقـــى (Relational Calculus) - التى تعد إحدى لغات النموذج العلاقى غير الإجرائية (Nonprocedural Language).

بعد التعرف على النمذجة المفاهيميـــة للبيانات (فى الفصل الثانى والثالث) وعلى النموذج العلاقى ولغاته الرســـمية (فى الفصل الرابع)، يشرح **الفصل الخامس** مرحلة التصميـــم المنطقى لقواعـــد البيانات. وتعتمد هـــذه المرحلة على النمـــوذج التمثيلى (Representational Model) المســـتخدم، وهـــو النموذج العلاقى فى هذا الكتاب. وتتكون مرحلـــة التصميـــم المنطقى من خطوتين رئيســـيتين: فى الخطـــوة الأولى يتم تحويل النموذج المفاهيمى إلى نموذج قاعدة البيانات المستخدمة. ولأن هذا الكتاب يركز على قواعد البيانات العلاقية، فإن هذه الخطوة تعنى تحويل النموذج المفاهيمى إلى النموذج العلاقـــى. أما فى الخطوة الثانية فيتم تحســـين تصميم قاعـــدة البيانات الناتجة من عملية التحويل بحيث تحتوى على أقل قدر ممكن من البيانات المتكررة حتى يتم تجنب المشـــكلات التى قد تنتج عن عمليات التعديل على محتويات قاعدة البيانات. وتدعى هـــذه الخطوة بعملية «التطبيع» (Normalization). ويركز الفصل الخامس على الخطوة الأولى من مرحلة التصميم المنطقى.

يوضح **الفصل السادس**، فـــى جزئه الأول، مفهوم «الجداول جيـــدة البناء» (Well-Structured Relations) بشكل غير رسمى. وبناءً على مفهوم «الجداول جيدة البناء» يتم شـــرح مفهوم «التطبيع» (Normalization) الذى يمثل «طريقة رسمية» (Formal Method) واضحة المعالم لها أسسها النظرية التى تمكن من التعرف على جودة الجداول التى تم تصميمهـــا فى الخطوة الأولى من التصميم المنطقى لقاعدة البيانات. ثم يشـــرح هذا الجزء من الفصل الســـادس مســـتويات التطبيع الأكثر اســـتخداماً فى تصميم قواعد البيانات، وذلك ابتداءً من الشكل الطبيعى الأول وانتهاءً بالشكل الطبيعى الرابع.

أمـا الجزء الثانى من الفصل السـادس فيركز على مرحلـة التصميم المادى لنظم قواعد البيانات. وتهدف هذه المرحلة من تصميم قواعد البيانات إلى إنشـاء تصميم يُمَكِّـن من تخزين البيانات بشـكل يوفر الأداء المناسـب لنظـام إدارة قاعدة البيانات علـى اختلاف حجـم العمليات التى تنفذ عليها. ويعنى هـذا، وعلى خلاف التصميم المفاهيمى والتصميم المنطقى، أن التصميم المادى يوضح الكيفية التى سـتخزن وتعالج فيها البيانات لا على الكيفية التى يتم من خلالها التعرف على البيانات والعلاقات فيما بينها أو طريقة تمثيلها وفق النموذج العلاقى أو نماذج البيانات الأخرى.

خصص **الفصل السـابع** لشرح بعض مكونات «لغة الاستفسار البنائية» (Structured Query Language (SQL) التى تعد واحدة من أكثر لغات قواعد البيانات العلاقية انتشاراً. وقد تم تبنى هذه اللغة من قبل «معهد المقاييس الوطنى الأمريكى» (American National Standards Institute (ANSI) و«منظمـة المقاييـس الدوليـة» (International Standards Organization (ISO). ويمكن تقسيم تعليمات لغة الاستفسار البنائية إلى ثلاث مجموعات من التعليمـات (أو «اللغات الفرعية» (Three Sub-languages)) وهـى: مجموعة (أو لغة) تعريـف البيانات، ومجموعة (أو لغة) معالجة البيانات، ومجموعة (أو لغة) التحكم فى البيانات. ونظراً لأهمية لغة الاستفسـار البنائية فى قواعـد البيانات العلاقية، فإن هذا يستلزم شـرح مكوناتها الأساسية بشكل مسـتفيض يميل إلى الجانب التطبيقى حتى يتسنى فهم طريقة عمل تعليماتها. لذا فقد تم تخصيص الفصل السابع والفصل الثامن لشـرح المكونات الأساسـية للغة الاستفسـار البنائية. ويحتوى الفصل السابع على شـرح لمجموعة تعليمات (أو لغة) تعريف البيانات وعلى شرح للعبارات الأساسية فى تعليمة الاختيار (أو الاسـترجاع) التى تعد من أهم تعليمات لغة معالجة البيانات. ويُستخدم فى هذا الفصل، والفصل الثامن كذلك، نظام إدارة قاعدة بيانات أوراكل فى تنفيذ واختبار تعليمات لغة الاستفسار البنائية القياسية، وذلك لكون هذه البيئة واحدة من الأوسـع انتشـاراً بين المتخصصين فى تطوير التطبيقات المبنية على نظم قواعد البيانات، هذا بالإضافة إلى تشـابهها مع ما توفره نظم إدارة قواعد البيانات المعروفة الأخـرى على المسـتوى التجارى من بيئات مشـابهة لتنفيذ تعليمات لغة الاستفسـار البنائية. أما بالنسـبة للقراء الذين لا تتوافـر لديهم بيئة أوراكل، وحتى يتمكن هؤلاء أيضاً من تطبيق مفاهيم وتعليمات لغة الاستفسـار البنائية والاسـتفادة القصوى من محتويـات هذا الفصل والفصل الثامن، فيُسـتخدم فى هذا الفصل نظام إدارة قاعدة بيانات «أكسـس» (ACCESS)، بشـكل مقتضب، وذلك لإعطاء فكرة مبسـطة لطريقة التعامل مع هذا النظام ولتنفيذ تعليمات لغة الاستفسار البنائية. والسبب الوحيد وراء

استخدام نظام إدارة قاعدة بيانات أكسس هو توافره فى غالبية الحاسبات الشخصية فى وقتنا الراهن، ومن ثم يمكن القارئ من الاستفادة القصوى من محتويات الفصول المتعلقة بلغة الاستفسار البنائية من خلال التطبيق العملى.

يستكمل **الفصل الثامن** شرح مكونات لغة الاستفسار البنائية حيث خصص الجزء الأول منه لاستكمال شرح تعليمة الاختيار (أو الاسترجاع)، عندما تقوم التعليمة بالتعامل مع أكثر من جدول فى آن واحد، ولشرح بقية مجموعة تعليمات (أو لغة) معالجة البيانات. أما الجزء الثانى من الفصل فقد خصص لشرح مجموعة تعليمات (أو لغة) التحكم فى البيانات.

يتطرق **الفصل التاسع**، وبشكل مقتضب، إلى أربعة موضوعات متطورة فى نظم قواعد البيانات وهى: **المعاملات**، وقواعد البيانات الشيئية، وقواعد البيانات العلاقية - الشيئية، وقواعد البيانات الموزعة. تمثل المعاملات الوسيلة الرئيسية التى يتم من خلالها التفاعل مع قواعد البيانات من قبل المستفيدين، سواء بشكل تفاعلى مباشر أو من خلال برامج التطبيقات التى يقوم مطورو التطبيقات ببنائها. أما نموذج البيانات الشيئى فقد تم تطويره لسد الاحتياجات التقنية التى يتطلبها تطوير نظم التطبيقات المتعلقة بمكننة أعمال المنظمات ذات الصبغة غير التقليدية من حيث البيانات التى تتعامل معها، مثل استخدامها لتطبيقات «التصميم بمساعدة الحاسب الآلى» (Computer-Aided Design)، و«التصنيع بمساعدة الحاسب الآلى» (Computer-Aided Manufacturing)، والتجارب العلمية، و«نظم المعلومات الجغرافية» (Geographical Information Systems)، و«تطبيقات الوسائط المتعددة» (Multimedia Applications)؛ على سبيل المثال لا الحصر. ونظراً لانتشار النموذج العلاقى وسهولته فى تمثيل البيانات والتعامل معها فقد عكف الكثير من الشركات المصنعة لنظم إدارة قواعد البيانات العلاقية على تبنى بعض مفاهيم النموذج الشيئى ضمن منتجاتها حتى تتمكن من مواكبة احتياجات المنظمات التى تتصف بياناتها بالصبغة غير التقليدية بالإضافة إلى تلك التى تتصف بالتقليدية. وأصبحت مثل هذه المنتجات تسمى قواعد البيانات العلاقية - الشيئية. أما بالنسبة للمنظمات التى تتوزع مقارُّها فى مناطق عديدة، وعلى رقع متباعدة جغرافياً فى الكثير من الأحيان، فقد دفعت هذه المنظمات الباحثين إلى تبنى مفهوم النظم الموزعة وأضحت تسمى فى مجال نظم قواعد البيانات «نظم قواعد البيانات الموزعة». وتوفر مثل هذه النظم العديد من الميزات مقارنة بتلك النظم المركزية من ضمنها «الموثوقية» (Reliability) و«التواجد» (Availability) هذا بالإضافة إلى أدائها المتميز وسهولة التوسع فى الأجهزة والتطبيقات فى مثل هذه المنظمات.

ونظراً لأهمية المفاهيم السابقة كان من الضرورى التطرق إليها فى هذا الكتاب ولو بشكل مقتضب.

يحتوى **الملحق رقم** (١) على حالة دراسية تمثل نظاماً مفترضاً للتسجيل فى إحدى الجامعات الأهلية. ويلخص الملحق عملية النمذجة المفاهيمية، والتصميم المنطقى، والتصميم المادى لقاعدة بيانات النظام. وبذلك فإن هذا الملحق يمثل تطبيقاً للعديد من المفاهيم الواردة فى الكتاب فى محاولة للاستفادة القصوى منه.

أما **الملحق رقم** (٢) فقد خصص لبعض التمارين التطبيقية على لغة الاستفسار البنائية؛ كونها من أهم مكونات نظم قواعد البيانات العلاقية. وتتفاوت صعوبة هذه التمارين من البسيطة جداً وحتى الصعبة جداً. وتهدف هذه التمارين فى مجملها إلى اختبار قدرة القارئ واستيعابه لمكونات لغة الاستفسار البنائية، من جهة، وإلى زيادة تمكنه من اللغة، من جهة أخرى.

وبناءً على الاستعراض السريع لمحتويات هذا الكتاب، يتضح أنه قد خصص، وبشكل رئيسى، للطلبة الدارسين فى مواد نظم قواعد البيانات من المتخصصين فى مجال الحاسب الآلى سواء فى مرحلة الدراسة الجامعية (البكالوريوس) أو طلبة الدبلوم (فوق الثانوى). كما أنه يستهدف أيضاً مطورى نظم التطبيقات الذين يتعاملون مع نظم قواعد البيانات العلاقية بشكل تطبيقى فى حياتهم اليومية، ليكون مرجعاً تطبيقياً لهم. وقد روعى فى الكتاب استخدام المصطلحات الإنجليزية مع ما يقابلها فى العربية؛ حتى يتمكن القارئ من فهم هذه المصطلحات فى حال عدم دقة الترجمة للمصطلح الإنجليزى، وحتى يتمكن من الربط بينهما عند الرجوع إلى أى مطبوعات أخرى مكتوبة باللغة الإنجليزية.

ويأمل المؤلف أن يلاقى هذا الكتاب قبولاً واستحساناً من الفئة المستهدفة من القراء، كما يتطلع إلى مقترحاتهم وآرائهم وانتقاداتهم البناءة التى من شأنها أن تثرى محتويات هذا الكتاب فى أية طبعة مستقبلية، سائلاً المولى عز وجل أن ينفع به ويفيد منه... إنه السميع العليم،،،

المؤلف

يوسف بن جاسم الهميلي

houmaily@ipa.edu.sa

ysf_al@yahoo.com

الفصل الأول

تطوير نظم المعلومات

صممت الحاسبات الآلية عند بداية ظهورها لحساب الدوال الرياضية المستخدمة فى التطبيقات العلمية والعسكرية التى تستنزف من الوقت الكثير لحسابها يدوياً. وكان ذلك بعد انتهاء الحرب العالمية الثانية. ولقد بدا واضحاً منذ بداية ظهور الحاسبات الآلية أهميتها فى حفظ ومعالجة البيانات إلا أنها لم تستخدم فى التطبيقات الإدارية للمنظمات المختلفة حتى بداية الخمسينيات الميلادية. ومن أسباب ذلك غلاء أسعار الحاسبات الآلية فى تلك الفترة وندرة المتخصصين فى التعامل معها. لذا فقد انحصر استخدام الحاسبات الآلية على فئات محدودة من المختصين الذين تتطلب طبيعة أعمالهم قدرات حسابية عالية. وكانت التطبيقات فى ذلك الوقت تتصف بمعالجتها لبيانات كثيرة وبشكل متكرر، ولذلك كان استخدام الحاسب الآلى مبرراً فى مثل هذه التطبيقات. ومع التطور المستمر والحثيث للحاسبات الآلية بالإضافة إلى التقلص المستمر فى أسعارها وزيادة أعداد المتخصصين فى التعامل معها، أصبحت الحاسبات تستخدم بشكل مكثف فى غالبية المنظمات الحديثة لما تقدمه من ميزات جمة مقارنة بالنظم اليدوية.

واستخدمت فى بداية ظهور الحاسبات الآلية الملفات لتصبح وسيلة رئيسية لتخزين البيانات. وكان كل نظام تطبيقى يتعامل مع الملفات الخاصة به فقط. إلا أن الزيادة المطردة فى أحجام البيانات وضرورة المشاركة فى استخدامها من قبل تطبيقات مختلفة، وذلك نتيجة طبيعية لزيادة أعداد المستفيدين من الحاسبات الآلية وتطور نظم التطبيقات؛ أدى إلى ظهور ما يعرف اليوم بنظم قواعد البيانات. وتتميز النظم المبنية على قواعد البيانات عن النظم المبنية على الملفات بعدد من المزايا الجوهرية التى أدت إلى تطور نظم قواعد البيانات بحيث أضحى هذا المجال واحداً من أكثر المجالات العلمية بحثاً وتطبيقاً فى وقتنا الراهن.

ويستعرض هذا الفصل الميزات التى تتحلى بها نظم قواعد البيانات فى بناء وتطوير نظم التطبيقات مقارنة بالطريقة التقليدية لتطوير النظم المعلوماتية. كما يستعرض هذا الفصل الخطوات الرئيسية المتبعة فى تحليل، وتصميم، وبناء، وإدارة قواعد

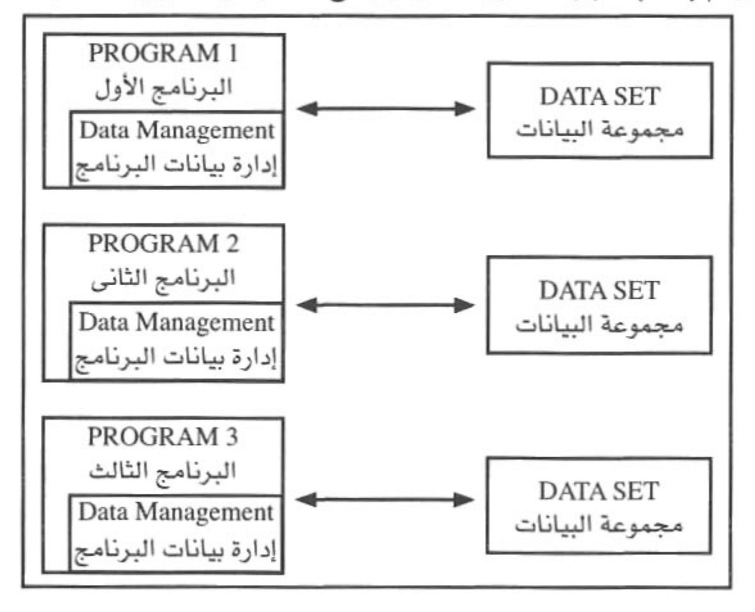
البيانـــات. ولأن قاعـــدة بيانات أى نظام معلوماتى تمثل جـــزءاً من النظام المعلوماتى، فـــإن هذا الفصل يبين موقع عملية تطوير قاعدة البيانات ضمن عملية التطوير الكلى للنظـــام المعلوماتى. ويلخص هذا الفصل فى نهايته أهم الأحداث التاريخية فى تطور نظم قواعد البيانات.

١-١ ميزات النظم التطبيقية المبنية على قواعد البيانات:

١-١-١ تطوير النظم باستخدام الملفات التقليدية:

عنـــد بداية ظهور الحاســـبات الآلية فى معالجة البيانات لـــم يكن هناك ما يعرف بقاعدة البيانات، وإنما كانت مجموعة البيانات المتعلقة ببرنامج حاســـوبى معين تدخل إلى الذاكرة الرئيسية للحاسب الآلى بشكل مباشر، وفى الوقت نفسه الذى تدخل فيه تعليمات البرنامج الذى سيقوم بمعالجتها، كما هو موضح بالشكل رقم (١-١).

شكل رقم (١-١): ارتباط البيانات بالبرامج عند بداية ظهور الحاسبات الآلية



وعنـــد انتهاء تنفيذ البرنامج تطبع نتائج معالجـــة البيانات، التى قام بها البرنامج، ويتم مســـح البرنامج مع بياناته من الذاكرة الرئيســـية للحاسب الآلى. ويعنى هذا أنه لـــم يكن موجوداً فى تلك الفترة ما يعرف بالقـــرص الصلب، أو الذاكرة الثانوية، الذى

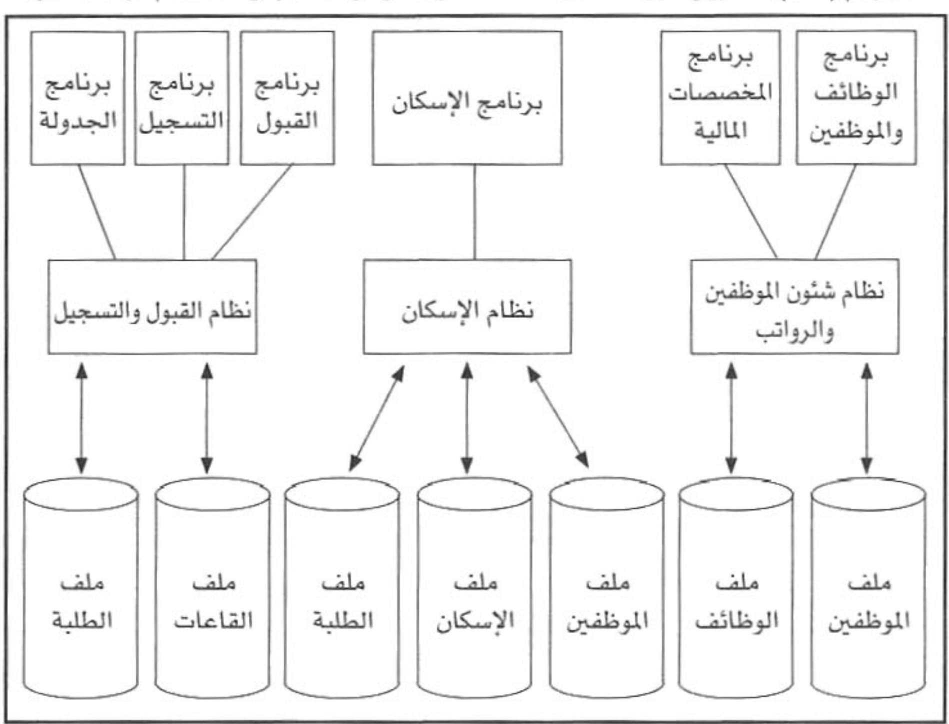
يحتوى على بيانات البرنامج بشكل دائم، وإنما كانت تدخل البرامج مع بياناتها فى كل مرة يتم فيها تنفيذ البرنامج. ومع تطور تطبيقات الحاسب الآلى وكبر حجم البرامج التى تتطلبها هذه التطبيقات بالإضافة إلى كبر حجم البيانات التى تقوم بمعالجتها، أصبح من المتعذر إدخالها فى الذاكرة الرئيسية للحاسب الآلى. بالإضافة إلى ذلك فإنه لم يكن معروفاً فى تلك الفترة ما يعرف بمبدأ المشاركة بين المستفيدين للحاسب الآلى، وإنما كان الحاسب الآلى مسخراً لخدمة برنامج واحد فى كل مرة. وعند انتهاء برنامج ما يدخل برنامج آخر مع بياناته، وهكذا.

ونتيجة لهذا القصور ظهرت الذاكرة الثانوية بشكل عام، والقرص الصلب على وجه الخصوص. كما ظهرت نظم التشغيل التى تسمح بمبدأ المشاركة. وسمح هذا التطور بتخزين البيانات فى ملفات على القرص الصلب للحاسب الآلى. ومع هذا التطور بدأ الحاسب الآلى بالانتشار بشكل واسع حتى فى التطبيقات الإدارية والمالية لمعظم المنظمات. وباستخدام هذا النمط الحاسوبى كانت التطبيقات تجزأ إلى مجموعة من البرامج بحيث يقوم كل برنامج بوظيفة واحدة أو مجموعة من الوظائف المترابطة منطقياً فيما بينها، كما كان لكل تطبيق مجموعته الخاصة من الملفات التى تحتفظ ببياناته على القرص الصلب. ويمثل الشكل رقم (١-٢) ثلاثة تطبيقات إدارية ومالية لإحدى المنظمات التعليمية الافتراضية. يُعنى النظام الأول، وهو نظام القبول والتسجيل، بقبول الدارسين وتسجيلهم فى المواد الدراسية المختلفة بالإضافة إلى جدولة القاعات الدراسية المتوافرة فى المنظمة. وتم تقسيم النظام إلى ثلاثة برامج فرعية، كل واحد منها يعنى بمجموعة من الوظائف المترابطة منطقياً فيما بينها. كما تم تعريف ملفين رئيسيين للنظام: أحدهما لتخزين البيانات الخاصة بالطلبة، والثانى لتخزين البيانات المتعلقة بالقاعات الدراسية.

أما النظام الثانى فقد خصص لإسكان أعضاء هيئة التدريس وإسكان الدارسين ويتكون من برنامج واحد. كما أن هذا النظام مرتبط بثلاثة ملفات رئيسية: أحدها لتخزين البيانات الخاصة بالطلبة، والثانى لتخزين البيانات المتعلقة بالمساكن المتوافرة وبيانات ساكنيها، والثالث لتخزين البيانات المتعلقة بالموظفين.

النظام الثالث فى هذه المنظمة هو نظام شئون الموظفين والرواتب الذى يتكون من برنامجين فرعيين: أحدهما يعنى بالوظائف المتوافرة فى المنظمة والموظفين المعينين عليها، أما الثانى فيعنى بالمخصصات المالية المختلفة للموظفين. ويرتبط هذا النظام بملفين: أحدهما مخصص للوظائف المتوافرة، والثانى مخصص لبيانات الموظفين.

شكل رقم (١-٢): تخزين البيانات فى ملفات على القرص الصلب واستخدام مبدأ المشاركة



ولا شك أن النظم السابقة نظم جيدة تؤدى الأهداف التى صممت من أجلها، إلا أن هذه النظم تعانى مشاكل جوهرية؛ لأنها ترتكز فى تصميمها وبنائها على الملفات التقليدية. ويمكن تلخيص هذه المشكلات فيما يلى:

- الاعتمادية (أو الترابط) بين البرامج والبيانات (Program-Data Dependence):

باستخدام الملفات التقليدية يجب توصيف الملفات المستخدمة لتخزين البيانات داخل برامج النظام التطبيقى، فعلى سبيل المثال يجب وصف ملف القاعات الدراسية وملف الطلبة فى برنامج الجدولة، كما يجب وصف ملف الطلبة فى كل من برنامج القبول وبرنامج التسجيل. ويقصد بوصف البرنامج تحديد طبيعة الملف إذا كان «متسلسلاً» (Sequential File) أو ذا «استرجاع عشوائى» (Random Access File). كذلك يقصد بوصف الملف تعريف البيانات التى ستخزن عليه من حيث مسمياتها ونوعية بياناتها (أعداد صحيحة (Integers)، أعداد «حقيقية» (Real (or Float))، «سلاسل حرفية» (Character Strings)، إلخ).

ويقصـــد بالاعتمادية بين البرامـــج والبيانات أن كل برنامج يجـــب أن يحتوى على توصيـــف للملفات التى يتعامل معهـــا من حيث نوعيتها، وتركيبهـــا، وطريقة الوصول إلى البيانات المخزنة عليها. وعند وجود ما يســـتدعى تغيير مكونات أحد الملفات فإن هـــذا يتطلب إعادة توصيف الملف قيد التغيير فى كافة البرامج التى تتعامل مع الملف. ولنفترض على ســـبيل المثال أنه وَجَب تغيير طول حقل مســـمى اسم العائلة للطلبة من عشرة حروف إلى خمسة عشر حرفاً نتيجة لقبول أحد الطلبة الذى يشكل اسم عائلته أكثر من عشـــرة حروف. فى هذه الحالة يجب تغيير سجل توصيف ملف الطلبة ليس فى برنامج القبول فحسب وإنما فى برنامج التسجيل وبرنامج توزيع القاعات الدراسية أيضاً؛ لأن كلاً من هذه البرامج يتعامل مع ملف الطلبة.

- **تكرارية البيانات (Duplication of Data):**

يتـــم تطوير النظم باســـتخدام الملفات التقليدية عادةً كلٌّ علـــى حدة، مما يعنى أن القاعدة الرئيســـية فى مثل هذه النظم هو تكرارية البيانات. فعلى سبيل المثال يحتوى ملف الإســـكان علـــى بيانات بعض الطلبة وبيانات بعض أعضـــاء هيئة التدريس، وهم الذيـــن يمثلون مجموعة الســـكان من ضمـــن المجموعة الكلية للطلبـــة وأعضاء هيئة التدريس. وينجم عن هذه التكرارية، فى غالبية الأحيان، ضياع كبير للطاقة التخزينية للحاســـبات. ليس هذا فحســـب وإنما ينجم عن هذا ضياع «لتكامل البيانات» (Data Integrity) التـــى يقصـــد بها إما عدم توافق «هيئة البيانـــات» (Data Format) المخزنة أو عدم توافق القيم المخزنة للبيانات نفسها أو الاثنين معاً. فعلى سبيل المثال لنفترض أن رقم الهاتف معرف على أنه «سلسلة حرفية» (Character String) مكونة من سبعة حروف فـــى ملف الطلبة التابع لنظام القبول والتســـجيل وملف الموظفين التابع لنظام شـــئون الموظفين والرواتب، وأن هذا الحقل نفسه معرف على أنه عدد «صحيح» (Integer) فى كل من ملف الطلبة وملف الموظفين التابعين لنظام الإســـكان. يمثل عدم التوافق هذا فـــى تعريف الحقل نفســـه ضياعاً لتكامل البيانـــات؛ إذ إن هيئة الحقل يجب أن تكون واحـــدة بغض النظر عن البرنامج الذى يتعامل معـــه. ولإيضاح الجانب الثانى لضياع تكامل البيانات لنفترض انتقال أحد الطلبة من مسكن إلى آخر الأمر الذى قد يتطلب تغييـــر رقـــم هاتف الطالب وعنوانه البريدى. ولأن ملـــف الطلبة يتبع لنظام آخر وهو نظام القبول والتســـجيل، فإن تغيير رقم هاتف الطالب وعنوانه البريدى قد يتم ضمن نظام الإســـكان ولكن القيم القديمة لرقم هاتف الطالب وعنوانه البريدى ســـتبقى كما هـــى ضمن ملف الطلبة التابع لنظام القبول والتســـجيل. ويعنـــى هذا ضياعاً لتكامل البيانات من حيث القيم المخزنة فيها.

- **المحدودية فى مشاركة البيانات (Limited Data Sharing):**

لكل نظام ملفاته الخاصة به عند استخدام طريقة تطوير النظم بالملفات التقليدية. ويعنى هذا أن المستفيدين من نظام معين ليس بإمكانهم التعامل مع البيانات المخزنة ضمن ملفات التطبيقات الأخرى بشكل مباشر؛ الأمر الذى تتطلبه طبيعة العمل فى الغالبية العظمى من الأحيان. فعلى سبيل المثال لنفترض أنه قد وَجَب إجراء خصم معين من مخصصات أعضاء هيئة التدريس كافة الذين تتوافر لهم مساكن من خلال إدارة الإسكان بالمنظمة. فى هذه الحالة يجب إجراء تعديلات على برنامج المخصصات المالية التابع لنظام شئون الموظفين والرواتب. ويستلزم مثل هذا الإجراء توصيف ملف الإسكان ضمن برنامج المخصصات المالية للموظفين، الذى يعد إجراءً عادياً فى مثل هذه البيئة الحاسوبية. ولكن المشكلة الحقيقية التى تظهر بوضوح فى مثل هذه الحالة هو عدم توافق ملف الإسكان مع بقية ملفات نظام شئون الموظفين والرواتب مثل تعريف طول اسم الموظف فى نظام الإسكان مع تعريف طول اسم الموظف فى نظام شئون الموظفين والرواتب. وتحد مثل هذه المشكلة من مشاركة البيانات بين التطبيقات المختلفة بسبب طول الفترة الزمنية اللازمة لكتابة برامج جديدة أو تعديل ما هو متوافر منها للتغلب على مشكلة عدم التوافق. وحتى لو سخر الوقت اللازم لكتابة البرامج اللازمة أو إجراء التعديلات على ما هو متوافر منها فإن مثل هذا الإجراء قد يضحى بسرية البيانات التى قد تكون هاجساً كبيراً فى بعض المنظمات. فعلى سبيل المثال لو عدل نظام الإسكان عوضاً عن نظام شئون الموظفين والرواتب للقيام بالخصم اللازم من أعضاء هيئة التدريس، فإن مثل هذا الإجراء قد يضحى بسرية المخصصات المالية للموظفين بحيث إن العاملين بإدارة الإسكان قد يكون بإمكانهم الاطلاع على مثل هذه البيانات.

ونتيجة لما سبق، فإن مبدأ مشاركة البيانات بين التطبيقات المختلفة يكون محدوداً جداً فى النظم التى تعتمد فى بنائها على الملفات التقليدية.

- **التطويل فى عملية التطوير (Lengthy Development Times):**

لكون كل تطبيق له ملفاته الخاصة التى تحتوى على بياناته فإنه توجد إمكانية محدودة للاستفادة من المجهودات السابقة التى بذلت فى تصميم نظم أخرى، إذ إن تطوير التطبيقات يبدأ عادة من نقطة الصفر بحيث يجب تصميم ملفات النظام الجديد متضمناً ذلك «هيئتها» (Format) وتوصيفها وبرمجة الطرق التى من خلالها

سيتم الوصول إلى البيانات. ولأن عملية تطوير تطبيقات جديدة لا تستفيد كثيراً من المجهودات السابقة؛ فإن هذا يؤدى إلى الإطالة فى عمليات التطوير حتى تصبح النظم الجديدة قيد الاستخدام التشغيلى الفعلى.

- **تكثيف صيانة البرامج (Excessive Program Maintenance):**

إن المساوئ السابقة مجتمعة تؤدى إلى تكثيف فى صيانة برامج التطبيقات. ومن أكثر أسباب تكثيف صيانة البرامج فى مثل هذه البيئة هو كون كل نظام مسئولاً عن عملية تهيئة بياناته وتخزينها وصيانتها بمعنى أن كل نظام مسئول عن تعريف الملفات الخاصة به وهيئة البيانات المخزنة فى كل ملف. وعند إجراء أى تعديل على تعريف السجلات، مثل إضافة أو حذف حقل ما أو حتى تغيير طول الحقل أو نوعية بياناته، فإن هذا يستدعى صيانة البرامج المسئولة عن قراءة البيانات من الملف والبرامج المسئولة عن كتابة البيانات عليه. ولأن عملية صيانة التطبيقات بشكل مكثف تستنزف مجهودات المسئولين عن هذه النظم من الفنيين، فإن هذا يحد بشكل كبير من الشروع فى تطوير تطبيقات جديدة تخدم أهداف المنظمة.

١-١-٢ تطوير النظم باستخدام قواعد البيانات:

يوفر استخدام نظم إدارة قواعد البيانات فى تطوير التطبيقات عدداً من الميزات على الطريقة التقليدية فى تطوير التطبيقات. ويمكن تلخيص هذه الميزات فيما يلى:

- **استقلالية البيانات عن برامح التطبيقات (Program-Data Independence):**

يتم وصف البيانات المخزنة فعلياً فى قاعدة البيانات بما يعرف «بالبيانات الوصفية» (Metadata). وتخزن البيانات الوصفية وأشياء أخرى تخص نظام إدارة قاعدة البيانات، بما يعرف «بالمستودع» (Repository). ويتم تطوير التطبيقات بمعزل عن البيانات الوصفية التى تصف البيانات الفعلية المخزنة فى قاعدة البيانات. لذلك فإن هنالك استقلالية بين برامج التطبيقات والبيانات المخزنة فى قاعدة البيانات، بمعنى أن نظم التطبيقات معزولة عن طريقة تمثيل البيانات وهياكلها فى قاعدة البيانات وكذلك طريقة تخزينها. لذا فإن نظم إدارة قواعد البيانات تسمح بتطور قاعدة البيانات ونموها دون أى إخلال أو تأثير فى التطبيقات التى تم تطويرها.

- **التحكم فى تكرارية البيانات (Controled Duplication of Data):**

تهدف نظـم إدارة قواعد البيانات إلى تجميع البيانات المخزنة فى ملفات منفصلة وإزالة أية ازدواجية فيها من خلال تمثيلها فى هيكل منطقى واحد، حيث يتم تسجيل كل حقيقة فى مكان واحد فقط. وعندما يلزم تكرار بعض البيانات الممثلة لحقائق معينة، فإن مثل هذا التكرار يكون محدوداً ومحكوماً من قبل نظام إدارة قاعدة البيانات.

- **تحسين تناسق البيانات (Improved Data Consistency):**

تحسن نظم إدارة قواعد البيانات من مستوى تناسق البيانات؛ إذ إنها تلغى ازدواجية البيانـات أو تتحكم فى الازدواجية عند وجودها. فعند تغيير أحد الموظفين أو الطلبة لعنوانه البريدي، على سبيل المثال، فإن هذا التغيير يتم من خلال تعديل بيانات الموظف أو الطالب فى مكان واحد فقط، ومن ثم فإن مثل هذا التعديل لا يؤدى إلى عدم تناسق للبيانـات مما قد يحدث عند اسـتخدام الملفات التقليدية التى تتكـرر فيها البيانات وتؤدى إلى عدم تناسقها. كما أن عملية التحديث تكون أيسر مقارنة بالملفات التقليدية لكونها تتم فى مكان واحد، ناهيك عن التغلب على المسـاحة التخزينية الضائعة نتيجة لتكرارية البيانات فى الملفات التقليدية.

- **تحسين مشاركة البيانات (Improved Data Sharing):**

إن جميـع بيانات المنظمة تخزن فى مكان واحد عند اسـتخدام نظم إدارة قواعد البيانـات عوضاً عن تخزين البيانات الخاصة بكل تطبيق فى ملفات مسـتقلة خاصة بالتطبيق. ويعنى هذا أن التطبيقات كافة تتشارك فيما بينها بالبيانات نفسها المخزنة فى قاعدة البيانات. وعلى الرغم من مشـاركة التطبيقات، والمستفيدين، لنفس قاعدة البيانـات، إلا أن نظام إدارة قاعدة البيانات يمكِّن من إعطاء المسـتفيدين صلاحيات محددة للتعامل مع البيانات المخزنة كل حسـب ما يحتاج إليه من بيانات جزئية تسمى «منظور المستفيد» (User's View).

- **تحسين إنتاجية نظم التطبيقات (Increased Productivity of Applications):**

يعد تحسـين الإنتاجية من نظم التطبيقات إحدى الميزات الرئيسية لطريقة تطوير التطبيقات باستخدام نظم قواعد البيانات، وذلك لسببين رئيسين:

١- بافتراض أنه قد تم تطوير التطبيقات التى تقوم بجمع البيانات وحفظها فى قاعدة البيانات فإن المبرمج لأى تطبيق جديد سـيقوم بالتركيز على برمجة الوظائف التى

سيقوم بها التطبيق قيد التطوير عوضاً عن التركيز على طريقة جمع البيانات وتخزينها فى الملفات وطريقة الوصول إليها.

٢- توفر غالبية نظم إدارة قواعد البيانات مجموعة من الأدوات التى تساعد على الإنتاجية، مثل أدوات «تصميم النماذج» (Form Design) وأدوات «توليد التقارير» (Report Generators)، بالإضافة إلى لغات عالية المستوى، مثل «لغة الاستفسار البنائية» (SQL) التى تمكِّن من التعامل مع قاعدة البيانات والوصول إلى البيانات المطلوبة بشكل فعال.

- **تقليص تكلفة صيانة البرامج (Reduced Program Maintenance):**

إن وصف البيانات والمنطق المستخدم للوصول إليها مبنى داخل التطبيقات نفسها عند استخدام الملفات التقليدية فى تطوير النظم، ولذلك فإن أى تعديل فى «هيئة البيانات» (Data Format) أو طريقة الوصول إليها يتطلب تعديل جميع التطبيقات التى تتعامل معها. إلا أن نظم قواعد البيانات توفر بعض الاستقلالية بين نظم التطبيقات والبيانات التى تستخدمها بمعنى أن تغيير هيئة البيانات أو طريقة الوصول إليها ليس من الضرورى أن ينعكس على جميع التطبيقات التى تتعامل مع هذه البيانات. كذلك هو الحال بالنسبة للتعديل على برامج التطبيقات، لأنه ليس من الضرورى أن ينعكس على البيانات المخزنة فى قاعدة البيانات. ولذلك فإن هذا يقلص من تكلفة صيانة التطبيقات مقارنة باستخدام الملفات التقليدية.

١-٢ نظم قواعد البيانات:

قاعدة البيانات هى مجموعة من البيانات التى نظمت بشكل متكامل؛ بهدف تلبية احتياجات عدد من المستفيدين فى المنشأة، للقيام بمهام أعمالهم. وتعد قاعدة البيانات أساس أى نظام معلوماتى سواء كان هذا النظام يدوياً أم آلياً. لذا فإننا نجد أن «نظم قواعد البيانات» (Database Systems) هى المرتكز الرئيس الذى تبنى عليه نظم المعلومات الآلية. وتتميز نظم قواعد البيانات عن «نظم الملفات» (File Systems) التقليدية التى كانت تستخدم لبرمجة التطبيقات مع بداية ظهور الحاسبات الآلية بالمميزات التالية (Efraim et al, 2002; Hoffer et al, 2002; Elmasri and Navathe, 2004):

- تكرار محدود للبيانات.
- عدم تعارض البيانات.

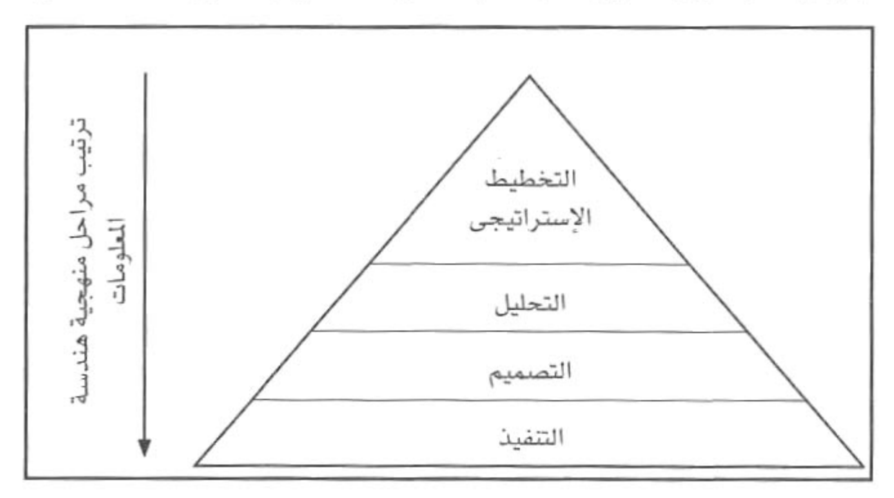
- تكامل البيانات.
- مشاركة البيانات.
- سهولة فرض «المقاييس» (Standards).
- زيادة إنتاجية المبرمجين.
- سهولة فرض ضوابط أمن وسلامة البيانات.
- الاستجابة للاحتياجات المتغيرة للبيانات.
- استقلالية البيانات.
- انخفاض صيانة برامج التطبيقات.

وفيما يلى شرح مقتضب لعملية تطوير نظم المعلومات باستخدام نظم قواعد البيانات.

١-٢-١ تطوير نظم المعلومات:

تعد منهجية «هندسة المعلومات» من أنجح المنهجيات فى تطوير نظم المعلومات؛ وذلك لكونها تعطى نظرة شاملة للمنشأة وتُمكن من تطويع تقنية المعلومات لخدمة أهدافها وسياساتها بكفاءةٍ عالية. ويعنى هذا تمكين المنشأة من التعرف على «عوامل نجاحها الحرجة» (Critical Success Factors) والعمل على مكننتها بهدف إعطائها الميزة التنافسية وتحسين أدائها. وترتكز منهجية هندسة المعلومات على أربع مراحل تأخذ شكلاً هرمياً (Martin, 1989)، كما هو مبين فى الشكل رقم (١-٣).

شكل رقم (١-٣): هرمية تطوير نظم المعلومات فى المنظمة وفق منهجية هندسة المعلومات



تأتى فى أعلى الهرم، الممثل فى الشـكل السابق، مرحلة **التخطيط الإستراتيجى** للمعلومات التى تهتم بالمعلومات الإسـتراتيجية للمنشأة، مثل أهداف المنشأة وعوامل نجاحها الحرجة. وتتمثل مرحلة التخطيط الإسـتراتيجى فى وضع خطة تكفل تكامل نظم المعلومات وجداولها الزمنية التى تحقق أهداف المنشأة. ويعنى هذا أن التخطيط الإسـتراتيجى لنظم المعلومات لا بد أن يرتبط بأهداف المنشـأة وأن يكون بعيد المدى لكى يتحقق نجاحه. كما يتم تحديد دور الإدارة العليا ومدى التزامها نحو تطوير نظم المعلومـات بحيث يجب على الإدارة العليا أن تتبنى نظم المعلومات عن قناعة وأن تلتزم بالتعامل معها باعتبارها أحد الموارد الأساسـية للمنشـأة. وبناء على ذلك يتم تحديد السياسـات والخطط والبرامج التى تضمن تنفيذ الخطة الإسـتراتيجية للمعلومات. وينتـج عن هـذه المرحلة تحديد للتطبيقـات التى تحتاج إليها المنشـأة وتوزيعها على «مناطق عمل» (Business Areas) حتى يسهل التعامل معها فى أثناء إجراء عملية التحليل المفصل والدقيق للمهام والبيانات. ويعنى هذا أن مرحلة التخطيط الإستراتيجى تمكن من إعطاء نظرة عامة عن المنشأة من حيث مهامها، وبياناتها، واحتياجاتها المعلوماتية، وهيكلها التنظيمى.

تلى مرحلة التخطيط الإستراتيجى للمعلومات **مرحلة التحليل** لأعمال القطاعات المختلفـة، حيث يتم فى هذه المرحلة النظر فى مهـام كل قطاع والبيانات الخاصة به. كما يتم فى هذه المرحلة تحديد الاحتياجات من التطبيقات وقواعد البيانات.

المرحلة الثالثة من منهجية هندسـة المعلومات هى **مرحلـة التصميم** حيث يتم فى هذه المرحلة تصميم النظم التطبيقية وقواعد البيانات للقطاعات المختلفة.

أما المرحلة الرابعة، والتى تمثل قاعدة الهرم فى منهجية هندسـة المعلومات، فهى **مرحلـة التنفيـذ** (أو الإنجـاز). فى هذه المرحلة يتم تحديد أشـكال البيانات وكتابة برامج التطبيقات.

تجدر الإشـارة هنا إلى أن المرحلتين الأولى والثانية تعتبران مسـتقلتين عن طبيعة الأجهـزة والنظم، فى حين أن المرحلتـين الثالثة والرابعة تعتمدان على طبيعة الأجهزة والنظم المسـتخدمة. ونظراً لأهمية مرحلة التخطيط الإسـتراتيجى لنظم المعلومات، فإننا نستعرض هذه المرحلة، بشكل مقتضب، فيما يلى.

١-٢-١-١ التخطيط الإستراتيجى لنظم المعلومات:

تهدف عملية التخطيط الإستراتيجى لنظم المعلومات إلى المواءمة بين التقنيات المعلوماتية المتوافرة وإستراتيجيات العمل المتبعة فى المنظمة. وتعد هذه المواءمة مهمة حتى تتم الاستفادة العظمى من الاستثمارات فى النظم المعلوماتية والتقنيات. وتمثل عملية التخطيط أساس عملية الانتقال من العمل اليدوى المعمول به فى المنظمة إلى العمل المميكن (أو المحوسب). وتتكون عملية (أو مرحلة) التخطيط فى منهجية هندسة المعلومات من ثلاث خطوات (Martin, 1989; Hoffer et al, 2002)، وهى كما يلى:

١- **تحديد عوامل التخطيط الإستراتيجى:** تتمثل عوامل التخطيط الإستراتيجى فى أهداف المنظمة، وعوامل نجاحها الحرجة، ومكامن (أو مصادر) المشكلات. وتهدف عملية تحديد هذه العوامل إلى تطوير إطار لعملية التخطيط يتم فيه الربط بين خطط النظم المعلوماتية وخطط العمل الإستراتيجية للمنظمة. ويحتوى الجدول رقم (١-١) على أمثلة لعوامل التخطيط الإستراتيجى الممكنة فى إحدى الجامعات الأهلية.

جدول رقم (١-١): أمثلة لعوامل التخطيط الإستراتيجى

عوامل التخطيط الإستراتيجى	أمثلة
الأهداف	زيادة عدد الخريجين بمعدل ٥٪.
	استقطاب كفاءات متخصصة ضمن أعضاء هيئة التدريس بالجامعة.
عوامل النجاح الحرجة	ارتفاع المستوى الأكاديمى لخريجى الجامعة.
	ارتفاع مستوى الإنتاج العلمى لأعضاء هيئة التدريس.
	زيادة إنتاجية أعضاء هيئة التدريس.
مكامن (أو مصادر) المشكلات	التوقعات غير الصحيحة لأعداد الطلبة المقبولين فى الجامعة.
	زيادة مستوى المنافسة مع الجامعات الأهلية الأخرى.

وتساعد العوامل السابقة إداريى نظام المعلومات فى وضع الأولويات التى تلبى متطلبات المستفيدين من النظم المعلوماتية الجديدة، ومن ثم تطوير قواعد البيانات فى المنظمة. فعلى سبيل المثال، مصدر مشكلة التوقعات غير الصحيحة لأعداد الطلبة المقبولين فى الجامعة قد يدفع إداريى النظام المعلوماتى إلى توفير المزيد من البيانات التاريخية عن الطلبة المقبولين فى الجامعة، ووضع المزيد من البيانات التى تنتج من

دراسات أعداد خريجى الثانوية العامة، وكذلك البيانات التى تنتج عن احتياجات سوق العمل من خريجى الجامعات.

٢- **التعرف على مكونات (أو وحدات) التخطيط:** التعرف على مكونات (أو وحدات) التخطيط فى مجال عمل المنظمة والذى يقوم بتقييد عملية تحليل النظم وتحديد المواقع التى تحدث فيها التغييرات. ويوجد خمسة مكونات (أو وحدات) للتخطيط، وهى كما يلى (Hoffer et al, 2002):

- **الوحدات التنظيمية (Organizational Units)** التى تتمثل فى الإدارات والأقسام المختلفة للمنظمة.
- **المواقع التنظيمية (Organizational Locations)** التى تتمثل فى المواقع المختلفة التى تدور فيها الفعاليات المختلفة لعمل المنظمة.
- **وظائف المنظمة (Business Functions)** التى تتمثل فى مجموعات من العمليات المرتبطة مع بعضها لمساندة مهام المنظمة. وتجدر الملاحظة إلى أن وظائف المنظمة تختلف عن الوحدات التنظيمية حيث أن وظيفة ما قد يتم القيام بها من قبل أكثر من وحدة تنظيمية واحدة. فعلى سبيل المثال، عملية تسجيل نتائج الدارسين فى إحدى الجامعات قد يتم القيام بها من خلال القسم الذى يتبعه كل دارس بالإضافة لإدارة (أو قسم) التسجيل فى إدارة القبول والتسجيل فى الجامعة.
- **أنواع الكينونات** التى تتمثل فى تصنيف للبيانات وفق الأشخاص، والأماكن، والأشياء التى تتعامل معها المنظمة أو تدار من قبلها.
- **النظم المعلوماتية** وتتمثل فى التطبيقات البرمجية والإجراءات التى تتعامل مع مجموعات من البيانات.

٣- **تطوير نموذج المنظمة (Developing an Enterprise Model):** يتكون النموذج المفصل للمنظمة من تفكيك وظيفى لكل وظيفة رئيسية تقوم بها المنظمة، ونموذج لبيانات المنظمة، ومصفوفات تخطيط مختلفة.

والتفكيك الوظيفى (Functional Decomposition) هو عملية تجزئة لوظائف المنظمة لمستويات أكثر تفصيلاً. وتعد عملية التفكيك الوظيفى من العمليات التقليدية التى تستخدم فى تحليل النظم حتى يتم تبسيط المشكلة، ويتم تركيز الانتباه على حلها، وللتعرف على مكوناتها.

أما نمذجة بيانات المنظمة فيتم توصيفها باستخدام ترميز معين مثل نموذج كينونة - علاقة، الذى يمثل محور الفصل الثانى والفصل الثالث من الكتاب. وبالإضافة للنموذج ذى الطابع الرسومى، يحتوى النموذج المفصل لبيانات المنظمة على توصيف لكل كينونة تمثل جزءاً من بيانات المنظمة، وعلى قواعد العمل المتبعة فى المنظمة التى تحكم صحة البيانات. وقواعد العمل هى عبارات لغوية تصف الأحكام والنظم واللوائح وأية اعتبارات مرعية أخرى تسير العمل فى المنظمة. كما يبين نموذج بيانات المنظمة أيضاً العلاقات التى تربط بين الكينونات المختلفة فى مخطط نموذج بيانات المنظمة.

أما العلاقات الأخرى بين وحدات التخطيط فيتم أخذها بعين الاعتبار أيضاً أثناء نمذجة المنظمة. ومن الأنماط المتعارف عليها فى تمثيل مثل هذه العلاقات مصفوفات وحدات التخطيط. وتقوم المصفوفات بتوفير وظيفة مهمة، هذه الوظيفة هى إظهار المتطلبات التى تحتاج إليها المنظمة من النظم المعلوماتية دون الحاجة إلى نمذجة قاعدة بيانات المنظمة التى يجب استخدامها مع هذه النظم. كما تساعد المصفوفات فى كثير من الأحيان فى وضع أولويات تطوير النظم المعلوماتية، وترتيب عمليات التطوير، وجدولة عمليات التطوير من خلال نظرة شاملة لاحتياجات المنظمة من النظم المعلوماتية. ويمكن استخدام عدد من مصفوفات التخطيط التى من بينها ما يلى:

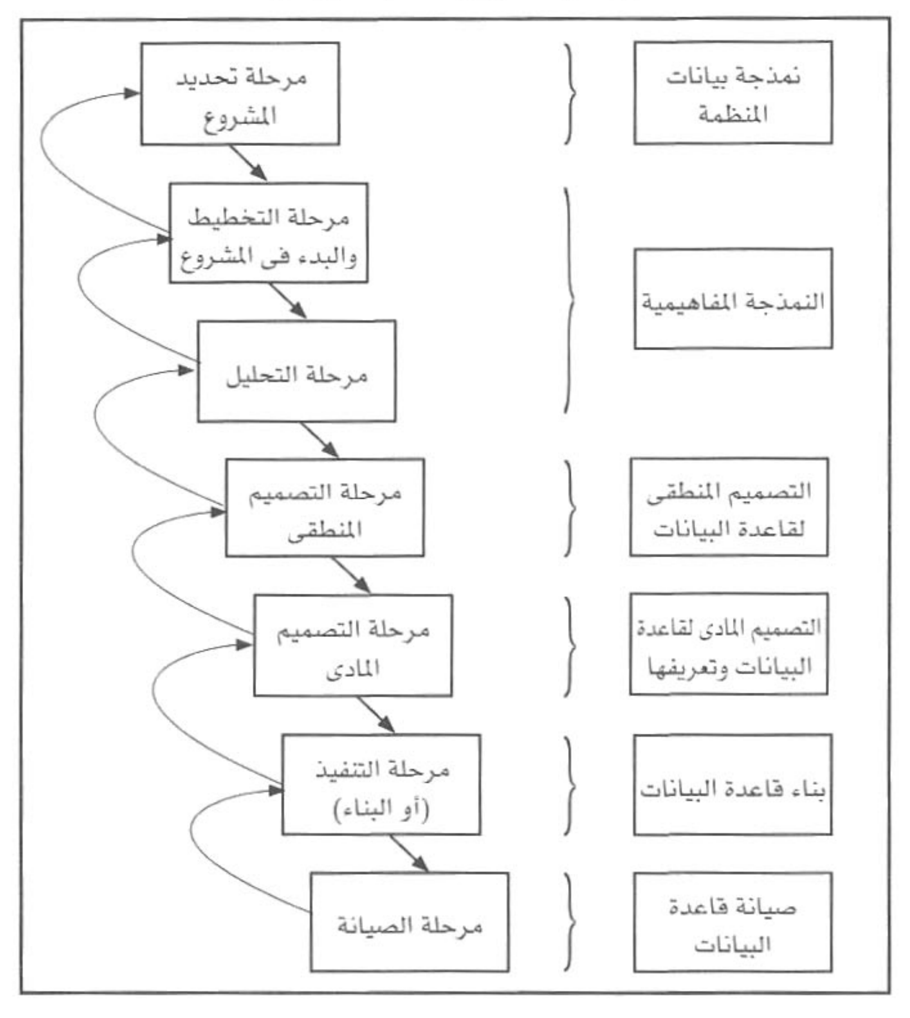
- **الموقع - الوظيفة:** يحدد هذا النوع من المصفوفات كل موقع يقوم بممارسة مهمة (أو وظيفة) ما.
- **الوحدة - الوظيفة:** يحدد هذا النوع من المصفوفات كل وحدة إدارية مسئولة عن (أو تقوم بممارسة) مهمة (أو وظيفة) ما.
- **النظام المعلوماتى - الكينونة:** يوضح هذا النوع من المصفوفات تفاعل كل نظام معلوماتى مع الكينونات التى تمثل بيانات المنظمة حيث توضح النظم المعلوماتية المسئولة عن إنشاء، أو استرجاع، أو تحديث، أو حذف كل نوع من أنواع البيانات المتوافرة فى المنظمة.

وبعد تحديد مناطق العمل والنظم المعلوماتية التى تحتاجها كل واحدة من مناطق العمل، يتم تطوير هذه النظم وفق المنهجية الأكثر انتشاراً وهى «دورة حياة النظم المعلوماتية»، التى نوجزها كما يلى.

١-٢-١-٢ دورة حياة تطوير النظم المعلوماتية:

تعـد منهجية تطوير النظـم المعلوماتية وفق ما يعرف بـ «دورة حياة تطوير النظم المعلوماتيـة» الطريقة التقليدية لتطوير النظم المعلوماتيـة. وتتكون هذه المنهجية من خطوات متكاملة يتم اتباعها لتوصيف، وتطوير، وصيانة، واستبدال النظم المعلوماتية. ويمكن تصور هذه المنهجية على أنها مجموعة من الخطوات التى تصب من واحدة إلى أخرى، كما هو مبين فى الجزء الأيسر من الشكل رقم (١-٤).

شكل رقم (١-٤): خطوات تطوير التطبيقات وفق منهجية «دورة حياة تطوير النظم المعلوماتية»



ويبين الجدول رقم (١-٢) الغرض (أو الهدف) من كل مرحلة ومخرجات المرحلة.

جدول رقم (١-٢): غرض كل مرحلة من مراحل «دورة حياة تطوير النظم المعلوماتية» ومخرجاتها

المرحلة	الغرض منها	مخرجاتها
تحديد المشروع	التعرف على دواعى تطوير نظام معلوماتى جديد أو تحسين نظام قائم.	اعتماد تصميم وتطوير نظام معلوماتى لتحسين أداء المنظمة فى جانب من الجوانب التى تقوم بها ضمن مهامها.
التخطيط والبدء فى المشروع	التعرف على الكيفية التى سيساهم فيها النظام المعلوماتى فى تحسين الوضع القائم.	اعتماد دراسة التغييرات الواجب إجراؤها على نظام قائم أو تطوير نظام معلوماتى جديد.
التحليل	التحليل التفصيلى للنظام بهدف تحديد المتطلبات وهيكلتها واختيار البدائل المناسبة لخصائص النظام.	مواصفات النظام الوظيفية التى تلبى احتياجات المستفيدين ويمكن تطويرها وبناؤها.
التصميم المنطقى	هيكلة كافة متطلبات النظام.	مواصفات وظيفية تفصيلية للبيانات والنماذج والتقارير والشاشات وقواعد معالجة البيانات.
التصميم المادى	تطوير المواصفات التنظيمية والتقنية فى المنظمة.	شراء التقنيات التى يتطلبها النظام. وتصميم هياكل البيانات وبرامج النظام.
التنفيذ (أو البناء)	كتابة برامج النظام، وبناء الملفات التى ستحتوى على بيانات النظام، وتركيب واختبار النظام، وتدريب المستفيدين، وتوثيق النظام.	برامج تعمل بشكل سليم وفق مواصفات النظام، ووثائق النظام، وأية مواد مستخدمة لتدريب المستفيدين.
الصيانة	مراقبة عمل وأداء النظام وفوائده، والعمل على إصلاح الأخطاء فيه، وتحسين أدائه.	المراقبة الدورية للنظام للتأكد من عمله بشكل سليم وتلبيته لاحتياجات المنظمة.

ويتشابه الشكل رقم (١-٤)، الذى يمثل دورة حياة تطوير النظم المعلوماتية، مع «الشلال المائى» (Waterfall)، إذ إن كل خطوة (أو مرحلة) تصب فى الخطوة (أو المرحلة) التالية. إلا أن هذه الخطوات قد لا تكون منفصلة تماماً على أرض الواقع: فبعض هذه الخطوات قد يتطابق فى بعض أجزائها من حيث وقت التنفيذ، وذلك فى حال إمكانية تنفيذها بشكل متزامن. كما أنه بالإمكان الرجوع إلى خطوات سابقة فى الشلال، فيما

يشبه التغذية المرتدة، وذلك عند ضرورة الرجوع إلى خطوات سابقة لإعادة النظر فى القرارات والاعتبارات التى تم اتخاذها فى تلك الخطوات.

ونظراً لأن تطوير قواعد البيانات يتم فى مراحل موازية لعملية تطوير النظم المعلوماتية: فإننا نلخص خطوات تطوير قواعد البيانات المقابلة لخطوات تطوير النظم المعلوماتية فيما يلى.

١-٢-١-٢-١ عملية تطوير قاعدة البيانات:

يوضح الشكل السابق (شكل رقم (١-٤))، الذى يمثل دورة حياة النظم المعلوماتية، فى جانبه الأيمن، الخطوات المتعلقة بتطوير قاعدة البيانات فى كل مرحلة من مراحل تطوير النظام المعلوماتى. وفيما يلى شرح لهذه الخطوات:

- **نمذجة المنظمة (Enterprise Modeling)**: تبدأ مرحلة تطوير قاعدة البيانات أثناء مرحلة نمذجة المنظمة، التى تعد جزءاً من مرحلة اختيار وتحديد المشروع. أما مرحلة نمذجة المنظمة، كما أسلفنا أعلاه، فتبدأ فى مرحلة التخطيط الإستراتيجى للمنظمة. وفى مرحلة نمذجة المنظمة، تتم مراجعة قواعد البيانات المتوافرة فى المنظمة، وتحليل طبيعة منطقة العمل التى يتبعها مشروع النظام المعلوماتى قيد التطوير، ويتم توصيف البيانات التى يحتاج إليها النظام بشكل عام دون الخوض فى تفاصيل البيانات.

- **النمذجة المفاهيمية (Conceptual Data Modeling)**: يتم فى هذه المرحلة تحليل المتطلبات العامة من البيانات التى يحتاج إليها النظام المعلوماتى. وتتم هذه العملية ضمن خطوتين: تتم الخطوة الأولى بالتزامن مع مرحلة التخطيط والبدء فى مشروع النظام المعلوماتى حيث يتم وضع تصور للبيانات التى يحتاج إليها النظام باستخدام نموذج كينونة – علاقة مبسط وتحديد قواعد البيانات الموجودة فعلياً وتحتوى على بيانات قد يتعامل معها النظام الجديد. ويتم توصيف البيانات فى هذه المرحلة والعلاقات فيما بينها بشكل مقتضب عالى المستوى دون الدخول فى التفاصيل الدقيقة لها. أما الخطوة الثانية فتتم بالتزامن مع مرحلة تحليل المشروع، وتهدف إلى إنتاج نموذج تفصيلى للبيانات يحدد كل البيانات التى يتعامل معها النظام. ويحتوى النموذج التفصيلى على جميع أصناف (أو فئات) البيانات، وجميع حقول أصناف البيانات، وتمثيل لكل العلاقات التى تربط البيانات بعضها ببعض، وتحديد كل قواعد العمل التى تعنى بتكامل (أو صحة) البيانات. ويحتوى الفصل الثانى من هذا الكتاب على شرح مفصل للمكونات الأساسية لنموذج البيانات كينونة – علاقة

(Entity-Relationship Model) الذى يعد أكثر نماذج البيانات المفاهيمية شيوعاً. أما الفصل الثالث فيحتوى على شرح لأهم مكونات نموذج كينونة - علاقة المطور الذى يساعد على النمذجة المفاهيمية للبيانات عندما تكون أكثر تعقيداً فى العلاقات فيما بينها.

- **التصميم المنطقى لقاعدة البيانات (Logical Database Design)**: يتم فى هذه المرحلة تحويل النموذج المفاهيمى إلى النموذج العلاقى بناءً على نظرية قواعد البيانات العلاقية وباستخدام شكل قياسى يسمى «العلاقات». وكلما تم تصميم أحد برامج النظام المعلوماتى، تتم مراجعة تفصيلية للعمليات التى تتفاعل مع قاعدة البيانات، والتقارير، والشاشات التى يحتويها البرنامج بهدف التعرف على كل البيانات التى يتفاعل معها النظام المعلوماتى وطبيعة هذه البيانات. وتوفر مثل عملية المراجعة هذه لبرامج النظام نظرة شمولية لقاعدة البيانات من الممكن أن تؤدى إلى إعادة النظر فى بعض جوانب النموذج المفاهيمى الذى تم تصميمه فى المرحلة السابقة وتعديلها حسب المعطيات الجديدة. وبعد ذلك تتم عملية تحويل مواصفات البيانات التى يتطلبها النظام إلى علاقات جيدة البناء تحتوى على أقل قدر من تكرارية البيانات التى تؤدى إلى أخطاء التعديل على قاعدة البيانات. وتستخدم فى هذه العملية قواعد مستمدة من نظرية قواعد البيانات العلاقية تسمى فى مجملها عملية «التطبيع» (Normalization). ويشرح الفصل الخامس من الكتاب التصميم المنطقى لقواعد البيانات العلاقية فى حين يشرح الجزء الأول من الفصل السادس مفهوم تطبيع العلاقات ومستويات تطبيعها.

- **التصميم المادى لقاعدة البيانات وتعريفها (Physical Database Design)**: فى هذه المرحلة يتم تحديد طريقة تنظيم قاعدة البيانات على وحدات تخزين الحاسب الآلى وهى الأقراص الصلبة عادة، كما يتم تعريف الهياكل المادية لإدارة قاعدة البيانات. وتهدف هذه المرحلة، بشكل عام، إلى التصميم الجيد لقاعدة البيانات على وحدات التخزين وتصميم الهياكل المناسبة لها بحيث يمكن للمستفيدين والنظم المعلوماتية أن تتعامل مع قاعدة البيانات بشكل فعال من حيث الأداء بالإضافة إلى تأمين السرية فى التعامل معها. ويعنى هذا أن التصميم المادى لقاعدة البيانات يجب أن يتم بتناسق كامل مع مراحل التصميم المادية الأخرى للنظام المعلوماتى، مثل برامج النظام، وبشكل يتوافق مع مكونات النظام الحاسوبى المستخدم، مثل نظام التشغيل، وشبكة الاتصالات المستخدمة. ويشرح الجزء الثانى من الفصل السادس بعض مفاهيم التصميم المادى والهياكل المستخدمة فى عملية التصميم المادى لقواعد البيانات.

- **بنـاء قاعـدة البيانـات (Database Implementation):** فى مرحلة بناء قاعدة البيانات تتـم كتابة، واختبار، وتركيب البرامج التى سـتتعامل مع قاعـدة البيانات. وقد تتم كتابة هذه البرامج باسـتخدام لغات البرمجة العامة (مثل كوبول، وسى، وجافا)، أو لغات معالجة خاصة بقواعد البيانات مثل «لغة الاستفسـار البنائية» (SQL)، أو أية لغة خاصة لكتابة التقارير وإظهار الشاشات والتى قد تحتوى على بعض الرسومات المعبرة. كما يتم فى هذه المرحلة إنهاء عملية توثيق تصاميم قاعدة البيانات وتدريب المستفيدين منها. أما الخطوة الثانية من هذه المرحلة فهى عملية تعبئة البيانات فى قاعـدة البيانات، وتتم هذه الخطوة من خلال تحويـل البيانات المتوافرة فى ملفات النظم قيد الاسـتخدام إلى صيغة قياسية (مثل النظام الثنائى أو ASCII أو Unified Code)، ثم تعبئتها فى قاعدة البيانات حسـب صيغة البيانات المستخدمة فى قاعدة البيانـات. أما فى حالة عدم توافر أية بيانات للنظام الجديد من نظم سـابقة قيد الاستخدام، فتتم عملية تعبئة البيانات حسب توافرها للنظام وإدخالها أولاً بأول.
- **صيانـة قاعـدة البيانـات (Database Maintenance):** فى مرحلة الصيانة تتم مراقبة قاعـدة البيانات، وذلـك لكونها تتصف بارتقائها وتطورها المسـتمر نتيجة لعمليات الحذف، والتعديل، والإضافة لهياكل بياناتها حتى تتناسب مع التغيرات المستمرة فى بيئة العمل؛ أو لتصحيح الأخطاء فى تصميم قاعدة البيانات أو لتحسين أداء معالجة البيانـات المخزنة فى قاعدة البيانات. ويمكـن النظر لهذه المرحلة على أنها مرحلة تطوير مقتضبة لقاعدة البيانات تحتوى على مراحل النمذجة المفاهيمية، والتصميم المنطقي، والتصميم المادى، والبناء حتى يمكن التعامل مع التغيرات المستمرة لقاعدة البيانـات. ويمكن النظـر لمرحلة الصيانة على أنها عمليـة تطوير لقاعدة البيانات ضمن فترات مراجعة يتم فيها النظر فى التغيرات الحادثة فى بيئة العمل وما تتطلبه هذه التغيرات من تطوير لقاعدة البيانات.

١-٢-١-٣ طرق التطوير البديلة لنظم المعلومات:

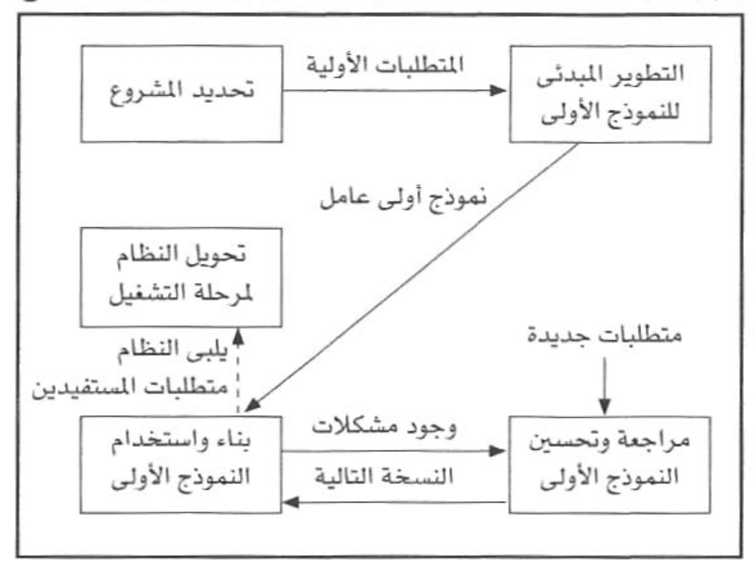
على الرغم من أن تطوير النظم المعلوماتية وفق منهجية «دورة حياة النظم المعلوماتية» التـى تحتوى على عدد من الخطوات و«نقاط التحقـق» (Checkpoints) لضمان تطوير النظم بدرجة عالية من الدقة والصحة فى نتائجها وتلبيتها لمتطلبات المستفيدين منها، إلا أن هذه المنهجية يتم انتقادها عادة بسـبب طول الفترات الزمنية التى تحتاج إليها لإنتاج النظم المعلوماتية. خاصة أن النظم التى يتم تطويرها وفق هذه المنهجية لا تكون

جاهزة للعمل إلا فى المرحلة الأخيرة من هذه المنهجية. ولهذا السبب فإن العديد من المنظمات تستخدم طرق تطوير بديلة لنظم المعلومات تتسم بالتطوير السريع لنظم التطبيقات. وتسمح هذه الطرق بالتطوير السريع للنظم من خلال عملية متكررة لمراحل التحليل، والتصميم، والبناء حتى يصبح النظام ملبياً لاحتياجات متطلبات المستفيدين. وتعمل طرق التطوير السريعة هذه أفضل ما يكون عندما تكون قواعد البيانات التى يتطلبها النظام قيد التطوير موجودة فعلياً على أرض الواقع، ومن ثم فإن النظام الجديد يتميز بكونه نظام استرجاع للبيانات عوضاً عن كونه نظام تحديث أو إدخال للبيانات ويتطلب مراجعة لهياكل قواعد البيانات.

١-٢-١-٣-١ النموذج الأولي (Prototyping):

تعد طريقة «النموذج الأولى» واحدة من أكثر طرق التطوير السريع لنظم المعلومات شيوعاً. وطريقة النموذج الأولى هى عملية تطوير للنظم المعلوماتية يتم فيها تحويل متطلبات المستفيدين إلى نظام عامل بحيث تتم مراجعته بشكل متكرر بين محللى النظام والمستفيدين منه حتى يتم الوصول إلى نسخة توافق متطلبات المستفيدين. ويوضح الشكل رقم (١-٥) خطوات تطوير النظم المعلوماتية وفق طريقة النموذج الأولى.

شكل رقم (١-٥): خطوات تطوير النظم المعلوماتية وفق طريقة النموذج الأولى



وبينما يوضح الشكل رقم (١-٥) مراحل تطوير النظم المعلوماتية وفق طريقة النموذج الأولى، فإن الجدول رقم (١-٣) يوضح الفعاليات المصاحبة لهذه المراحل والتى تتعلق بتطوير قاعدة البيانات بشكل تقريبى.

جدول رقم (١-٣): الفعاليات المتعلقة بقواعد البيانات المصاحبة لمراحل تطوير النظم المعلوماتية وفق طريقة النموذج الأولي

مرحلة تطوير النظام المعلوماتى	الفعاليات المتعلقة بقواعد البيانات
تحديد المشروع	نمذجة مفاهيمية: - تحليل متطلبات النظام. - تطوير نموذج مبدئى للبيانات التى يتطلبها النظام.
التطوير الأولى للنموذج المبدئى	١- التصميم المنطقى لقاعدة البيانات: - تحليل تفصيلى لمتطلبات النظام. - تحويل النموذج المفاهيمى إلى النموذج العلاقى. - تطبيع العلاقات. ٢- التصميم المادى لقاعدة البيانات: - تعريف محتويات قاعدة البيانات. - تنظيم محتويات قاعدة البيانات مادياً. - تصميم البرامج التى تتعامل مع قاعدة البيانات.
بناء واستخدام النموذج الأولى	بناء قاعدة البيانات: - كتابة التعليمات التى تتعامل مع قاعدة البيانات ضمن برامج النظام. - تعبئة البيانات فى قاعدة البيانات من مصادر البيانات المتوافرة (فى النظم القائمة).
مراجعة وتحسين النموذج الأولى	صيانة قاعدة البيانات: - تحليل قاعدة البيانات للتأكد من تلبيتها لمتطلبات النظام المعلوماتى. - تصحيح الأخطاء فى قاعدة البيانات.
تحويل النظام لمرحلة التشغيل	صيانة قاعدة البيانات: - تحسين أداء قاعدة البيانات. - تصحيح الأخطاء فى قاعدة البيانات.

تتم عملية النمذجة المفاهيمية لقاعدة البيانات، وبشكل مقتضب، عندما يتم تحديد النظام المعلوماتى الذى يجب تطويره. وفى مرحلة التطوير المبدئى للنموذج الأولى يتم تصميم الشاشات والتقارير التى تلبى احتياجات المستفيدين، وفى الوقت نفسه،

التعرف على أية متطلبات إضافية يحتاج إليها المستفيدون من قاعدة البيانات إضافة إلى تعريف قاعدة البيانات. وعادة ما تكون قاعدة البيانات الخاصة بالنظام قيد التطوير نسخة تتألف من أجزاء قواعد بيانات متوافرة فى المنظمة وقيد العمل بها، مع إمكانية احتوائها على بيانات جديدة.

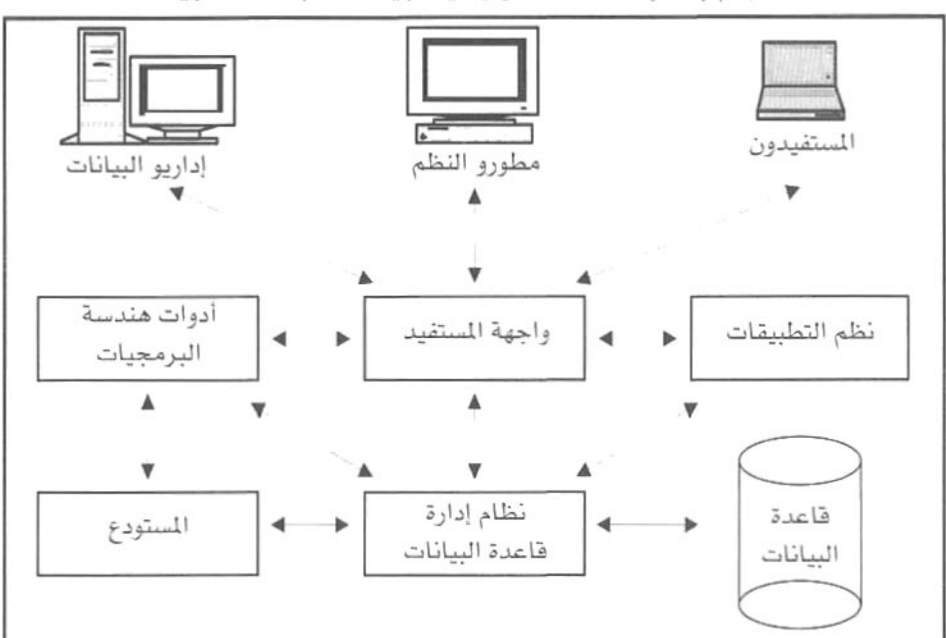
وبعد تطوير النموذج المبدئى والبدء فى استخدامه، ينتقل النظام إلى حلقة تتكرر فيها مرحلة البناء ومرحلة المراجعة. وعند بداية انتقال النظام لهذه الحلقة، تكون جوانب السرية والتكامل فى النظام أقل ما يمكن: لأن التركيز كان منصباً على الحصول على نموذج أولى عامل. وبعد أن يصل النظام إلى وضع مستقر يلبى غالبية متطلبات المستفيدين، نتيجة لمروره بعدد من مراحل المراجعة والبناء، ويقرر كل من المستفيدين ومطورى النظام نقله للمرحلة الإنتاجية (أو التطبيقية)، يدخل النظام فى مرحلة الصيانة ويبقى فى هذه المرحلة ما بقى النظام قيد الاستخدام.

وكما يلاحظ من خلال الشرح السابق، فإن طرق التطوير السريعة تمكن من الوصول إلى نظام عامل بأقل وقت ممكن ولكنه يلبى أقل قدر ممكن من احتياجات المستفيدين. أما منهجية دورة حياة النظم المعلوماتية فتلبى أكبر قدر ممكن من احتياجات المستفيدين، ولكنها تتطلب وقتاً كبيراً للوصول إلى نظام عامل. ولذلك فإن الاختيار بين الطريقتين يعتمد بحد كبير على طبيعة النظام المعلوماتى قيد التطوير، وخاصة درجة تلبيته لاحتياجات المستفيدين فى صورته الأولى، من جانب، والوقت المتوافر لعملية تطوير النظام، من جانب آخر.

١-٢-٢ مكونات بيئة نظام قاعدة البيانات:

تعد البيئة التشغيلية لنظام قاعدة البيانات بيئة متكاملة من الأجهزة والبرامج والأفراد الذين يشرفون على إدارة وتشغيل نظام قواعد البيانات. ويوضح الشكل رقم (١-٦) المكونات الرئيسية لبيئة نظام قواعد البيانات وعلاقتها ببعضها.

شكل رقم (١-٦): المكونات الرئيسية لبيئة نظام قاعدة البيانات



وفيما يلى توضيح للمكونات الرئيسية لبيئة قاعدة البيانات:

١- **أدوات هندسة البرمجيات المساعدة (Computer-Aided Software Engineering (CASE) Tools)**: هى أدوات آلية مصاحبة لنظام قاعدة البيانات تستخدم عادة فى عملية تصميم قواعد البيانات وبرامج التطبيقات.

٢- **مستودع (Repository)**: يُعَدُّ المستودع قاعدة معارف مركزية تحتوى على تعريف لجميع البيانات والشاشات والتقارير. كما يحتوى المستودع على معلومات عن جميع مخططات قاعدة البيانات والقيود عليها، وعلى معلومات أخرى يدخل من ضمنها القرارات التى تم اتخاذها فى عملية تصميم قاعدة البيانات، ومقاييس استخدام قاعدة البيانات، ووصف للتطبيقات التى ستتعامل مع قاعدة البيانات، ومعلومات عن المستفيدين من قاعدة البيانات. ويمكن الرجوع إلى هذه المعلومات عند الحاجة، سواء من المستفيدين من قاعدة البيانات أم من إداريى قاعدة البيانات. ويمكن تشبيه مستودع المعلومات بكتالوج نظام إدارة قاعدة البيانات (DBMS Catalog) الذى يحتوى على معلومات تفصيلية عن البيانات المخزنة فى قاعدة البيانات، وطرق استرجاعها

(Access Paths)، وأماكن وجودها، بالإضافة إلى معلومات عن حقوق المستفيدين فى التعامل معها (Users' Access Information). إلا أن المستودع يحتوى على معلومات أكثر تنوعاً من كتالوج نظام إدارة قاعدة البيانات كما أنه مسخر للتعامل معه بشكل مباشر من قبل المستفيدين من نظام قاعدة البيانات، من مبرمجين للتطبيقات وإداريين لقاعدة البيانات، عوضاً عن استخدامه من قبل برامج نظام إدارة قاعدة البيانات الذى هو حال كتالوج نظام إدارة قاعدة البيانات.

٣- **نظام إدارة قاعدة البيانات ((Data Base Mangement System (DBMS))**: هو نظام برمجى تجارى يستخدم لإدارة قاعدة البيانات من حيث تخزين واسترجاع وتحديث البيانات طبقاً لمتطلبات المستفيدين من قاعدة البيانات، كما أنه مسئول عن سلامة البيانات وتكاملها حتى فى ضوء التداول المتزامن لها من قبل المستفيدين من البيانات وفى حالة حدوث عطل من الأعطال للنظام الحاسوبى الذى يحتوى على قاعدة البيانات.

٤- **قاعدة البيانات (Database)**: هى مجموعة من البيانات المترابطة منطقياً فيما بينها تم تصميمها لتلبية الاحتياجات المعلوماتية لمجموعة من المستفيدين فى المنظمة. وتحتوى قاعدة البيانات على البيانات الفعلية الموجودة فى المنظمة عوضاً عن وصفها الذى يكون مخزناً فى المستودع.

٥- **برامج التطبيقات (Application Programs)**: هى مجموعة من البرمجيات تم تطويرها لتستخدم من قبل المستفيدين من قاعدة البيانات بحيث تمكنهم من إدخال البيانات فى قاعدة البيانات والتعديل عليها، بالإضافة لمعالجتها للحصول على معلومات تتناسب مع احتياجاتهم.

٦- **واجهة المستفيد (User Interface)**: هى مجموعة من اللغات والقوائم (Menus) والأوامر والنماذج (Forms) التى تمكن المستفيدين من التعامل مع بقية مكونات النظام وما فيها من أدوات هندسة البرمجيات، وبرامج التطبيقات، ونظام إدارة قاعدة البيانات، والمستودع.

٧- **إداريو قواعد البيانات ((Database Administrators (DBAs))**: هم الأشخاص المسئولون عن تصميم قواعد البيانات ووضع سياسات أمنها وسلامتها واستعادتها لوضعها التشغيلى الطبيعى بعد حدوث الأعطال. ويستخدم إداريو قواعد البيانات نظام إدارة قواعد البيانات وأدوات هندسة البرمجيات والمستودع فى أداء مهامهم.

٨- **مطورو النظم (Systems' Developers):** هم محللو النظم والمبرمجون الذين يقومون بتطوير نظم التطبيقات التى تحتاج إليها المنظمة. وعادة ما يستخدم مطورو النظم أدوات هندسة البرمجيات فى عملية تحليل احتياجات المستفيدين وتصميم النظم.

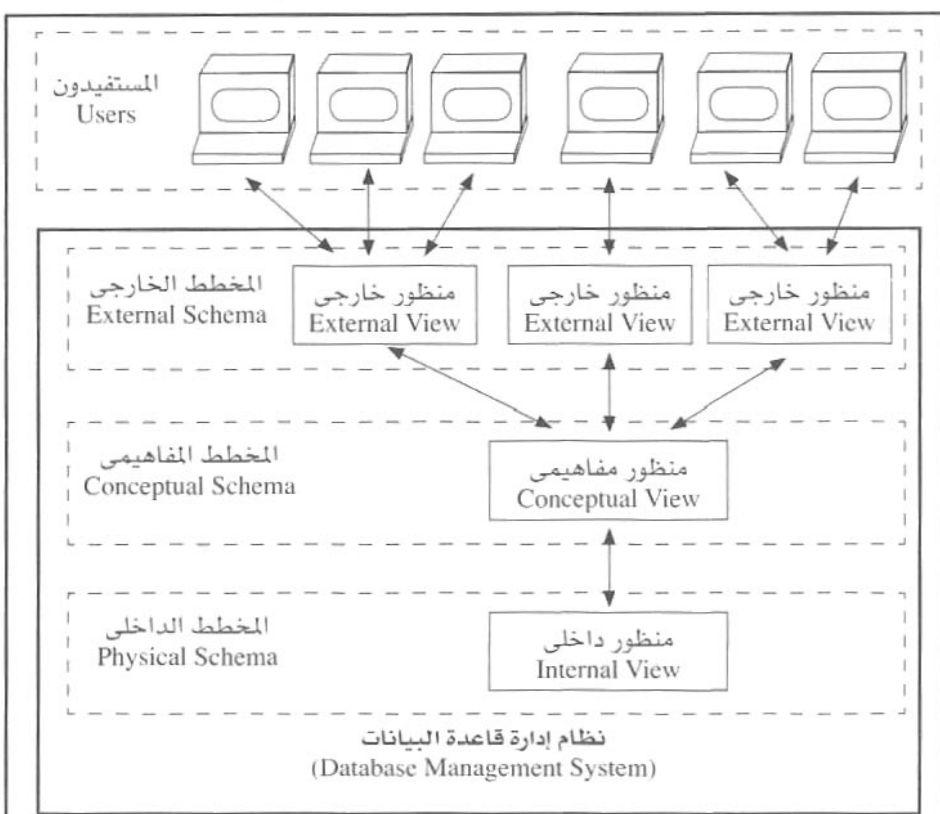
٩- **المستفيدون (End Users):** هم الأشخاص الذين يتعاملون مع قاعدة البيانات من خلال إدخال البيانات إليها أو إجراء التعديلات عليها، ويطلبون أو يستقبلون معلومات منها.

ويلاحظ فى الشكل السابق لبيئة نظام قاعدة البيانات أن جميع التعاملات مع قاعدة البيانات المخزنة فى النظام لا بد أن تمر من خلال نظام إدارة قاعدة البيانات (DBMS)، لكونه الجزء المسئول عن سلامة البيانات وتكاملها فى جميع الأحوال التشغيلية للنظام.

١-٢-٣ مستويات التجريد: المنظورات الثلاثة (Levels of Abstraction: Three-Schema Architecture):

إن الهدف من وراء بناء قواعد البيانات وفق المنظورات الثلاثة هو الفصل بين برامج التطبيقات التى يستخدمها المستفيدون والبيانات المخزنة فى قاعدة البيانات. وبهذه الطريقة يتم توصيف البيانات فى نظم إدارة قواعد البيانات وفق ثلاثة مستويات من التجريد. ويتكون توصيف البيانات فى كل مستوى من هيكل (Schema) للبيانات. ويوضح الشكل رقم (١-٧) مستويات التجريد الثلاثة وعلاقة كل منها بالأخرى. كما تجدر الإشارة هنا إلى أن المستويات الثلاثة كافة ما هى إلا توصيف لقاعدة البيانات نفسها، ولكنه يتم من خلال منظورات مختلفة.

شكل رقم (١-٧): مستويات التجريد فى بيئة نظام قاعدة البيانات



١- المنظور الداخلى (Internal View):

للمنظور الداخلى هيكل (Schema) يصف التركيبة الداخلية لقاعدة البيانات متضمناً ذلك تفاصيل تخزين البيانات، مثل مسميات الملفات المخزنة فيها، وأماكن تخزينها على الأقراص الصلبة، وعما إذا كانت البيانات متكررة على مجموعة من الأقراص الصلبة للتمكن من استعادتها فى حالة الأعطال، بالإضافة إلى طرق الوصول إليها.

٢- المنظور المفاهيمى (Conceptual View):

للمنظور المفاهيمى هيكل (Schema) يصف تركيبة كامل قاعدة البيانات لمجموعة من المستفيدين. ويخفى المنظور المفاهيمى تفاصيل تخزين البيانات فى المستوى الداخلى.

حيث يركــز على وصف الكينونات، والعلاقات، ونوعية البيانات (Data Types)، والقيود المفروضــة على قاعدة البيانـــات دون الخوض فى تفاصيل تخزيــن البيانات وطريقة الوصول إليها.

٣- المنظور الخارجى (External View):

يتكون المنظور الخارجى من مجموعة من الهياكل التى تصف منظورات المستفيدين مــن قاعدة البيانات. ولكل مجموعة من المســتفيدين منظورهـــا الخاص الذى يهمها مــن قاعدة البيانات. وكل هيكل خارجى يصف جزءاً من قاعدة البيانات يدخل ضمن اهتمام مجموعة محددة من المســتفيدين ويخفى ما تبقى من قاعدة البيانات عن هذه المجموعة.

تجدر الملاحظــة إلى أن المنظورات الثلاثة ما هى إلا وصــف لقاعدة البيانات: إذ لا يوجد ســوى قاعدة بيانات واحدة مخزنة فى المســتوى المادى. ولأن كل مســتخدم يتعامــل مع منظوره (الخارجى) لقاعدة البيانات، فإن نظام إدارة قاعدة البيانات يجب أن يترجم أى تعامل من قبل المستخدم لما يكافئه من تعامل على المنظور المفاهيمى، ومن ثم ما يكافئه على المنظور الداخلى.

ويمكن استخدام المنظورات الثلاثة لشرح المقصود باعتمادية البيانات بشكل أوضح والتى يقصد بها فى هذه الحالة القدرة على تغيير هيكل البيانات فى مســتوى ما دون الحاجة إلى تغيير هيكل البيانات فى المســتوى الــذى يعلوه. ويقودنا هذا إلى تعريف مستويين من عدم اعتمادية البيانات كما يلى:

١- **عـدم الاعتماديــة المنطقية:** هى القدرة علــى تغيير الهيكل المفاهيمى دون الحاجة إلى تغييــر المخطط الخارجى أو برامج التطبيقات. إذ يمكننا إضافة جدول جديد أو حقــل ضمــن جدول أو قيد جديد على قاعدة البيانـــات دون الحاجة إلى تغيير المخططـــات الخارجيــة أو التعديل على برامج التطبيقـــات. كما أنه يمكننا حذف جــدول أو حقل ضمن جدول دون الحاجة إلى تعديل المنظورات كافة أو التطبيقات كافــة، وإنما يكتفــى بتعديل تلك المنظورات والتطبيقات التى ســتتأثر بمثل عملية الحــذف هذه. كمــا يمكن حذف بعض القيود أو التعديــل عليها دون الحاجة إلى تعديل أى من المنظورات الخارجية أو أى من برامج التطبيقات.

٢- **عـدم الاعتماديــة المادية:** هى القدرة علــى تعديل الهيكل الداخلى لقاعدة البيانات دون الحاجة إلـــى تغيير الهيكل المفاهيمى، ومن ثم عدم الحاجة إلى تغيير الهياكل

الخارجية كذلك. ومن التغييرات التى قد تطرأ على الهيكل الداخلى إعادة ترتيب بعض ملفات قاعدة البيانات، مثل إنشاء هياكل جديدة للوصول إلى البيانات (Access Structures) التى تساعد على الوصول إلى البيانات بفاعلية أكبر.

١-٢-٤ أنواع قواعد البيانات المتوافرة على المستوى التجارى:

يتوافر حالياً فى الأسواق عدد كبير جداً من نظم إدارة قواعد البيانات التى تُراوح مجالات استخداماتها بين «الشخصية» (Personal Databases) القابلة للاستخدام الفردى، مثل تلك التى يمكن تركيبها على «الحاسبات الشخصية» (Personal Computers)، أو «المساعد الرقمى الشخصى» (Personal Digital Assistance)، وصولاً إلى تلك التى يمكن تركيبها واستخدامها على «الحاسبات المركزية» (Mainframes) التى قد يصل أعداد مستخدميها إلى الآلاف منهم. وعلى الرغم من الاختلافات الجوهرية، فى بعض الأحيان، فى التقنيات المستخدمة فى بناء نظم قواعد البيانات إلا أنها تجتمع فيما تقدمه من ميزات على نظم الملفات التقليدية بوصفها وسيلة لتخزين وإدارة البيانات، التى سبق أن تم استعراضها أعلاه. ويُعنى هذا الكتاب، وبشكل خاص، بنظم قواعد البيانات العلاقية؛ وذلك لكونها الأكثر انتشاراً فى وقتنا الراهن على الرغم من وجود نظم إدارة قواعد بيانات مبنية على نماذج أخرى للبيانات مثل «الهرمية» (Hierarchical) و«الشبكية» (Network) و«الشيئية» (Object-Oriented).

١-٣ سرد تاريخى لتطور نظم قواعد البيانات:

كانت بداية ظهور نظم إدارة قواعد البيانات فى الستينيات الميلادية، وهى منذ ذلك الحين تتطور بشكل مستمر. ويستعرض هذا الجزء من الكتاب أهم التطورات فى هذا المجال.

الستينيات الميلادية:

اعتمدت الغالبية العظمى من التطبيقات فى هذه الحقبة الزمنية على الملفات فى بنائها. إلا أن هذه الحقبة شهدت ظهور أولى نظم إدارة قواعد البيانات، وقد تم تصميمها من قبل شارلز باتشمان (Charles Bachman) الذى كان يعمل فى شركة جنرال إلكتريك (Ramakrishnan and Gehrke, 2003). وقد سمى هذا النظام بمخزن البيانات المتكامل (Integrated Data Store) الذى أصبح فيما بعد أساساً لنموذج البيانات الشبكى

(Network Data Model)، كمـا تم وضـع مقاييس لهذا النمـوذج من خلال مؤتمر عرف بمؤتمر لغات نظـم البيانـات ((Conference on Data Systems Languages (CODASYL)) الذى كان له أثر كبير فى مسـار نظم قواعد البيانات عبر الستينيات الميلادية. ولقد حصل باتشـمان على أول جائزة تورينغ (ACM Turing Award)، وهى جائزة الحاسـب الآلى المكافئـة لجائزة نوبل على أعماله فى مجال نظـم قواعد البيانات عام ١٩٧٣م (Ramakrishnan and Gehrke, 2003). وفى أواخر الستينيات الميلادية، طورت شركة أي. بي. إم. (IBM) نظاماً سُـمى بنظام إدارة المعلومات (Information Management System) (IMS) الذى أصبح فيما بعد أساساً لنموذج البيانات الهرمى (Hierarchical Data Model) الذى يعد نموذجاً بديلاً للنموذج الشبكى.

وكان استخدام نظم قواعد البيانات فى الستينيات الميلادية محصوراً فى التطبيقات التى تغلب عليها ضخامة البيانات وتعقد مهامها مثل التطبيقات التى اسـتخدمت فى مشروع هبوط مركبة الفضاء أبولو (Apollo) على سطح القمر (Hoffer et al, 2002). كما تميزت هذه الفترة أيضـاً ببداية المجهودات التى تهدف لوضع المقاييس المتعلقة بنظم إدارة قواعد البيانات من خلال تشـكيل فريق مهمـة قواعد البيانات (Data Base Task Group) مع نهاية العقد.

السبعينيات الميلادية:

أضحت قواعد البيانات فى هذا العقد واقعاً ملموسـاً على المستوى التجارى، فتم تطوير نظـم إدارة قواعد البيانـات «الهرمية» (Hierarchical) و«الشـبكية» (Network) للاستخدام فى تطوير التطبيقات ذات هياكل بيانات معقدة يصعب تطويرها باستخدام الملفات التقليدية. وتُمثل نظم إدارة قواعد البيانات الهرمية والشبكية التى طورت فى هذه المرحلة الجيل الأول لنظم إدارة قواعد البيانات على المسـتوى التجاري. وقد تم اسـتخدام كلا النموذجين بشكل كبير فى هذه المرحلة. وما زالت هنالك بعض النظم التى تستخدم هذين النموذجين قيد العمل والاستخدام حالياً.

وعلى الرغم من نجاح كلا النموذجين وانتشـارهما فى هذه المرحلة، إلا أن كليهما كان يعانى بعض المساوئ الجوهرية التى يمكن تلخيصها فيما يلى:

١- صعوبـة التنقل بين البيانـات: فالبيانات تخزن على هيئة «سـجلات» (Records) ووسـيلة التنقل المسـتخدمة تعتمد، فى كلا النموذجين، على الانتقال من سـجل

للبيانات إلى آخر. ويعنى هذا ضرورة كتابة برامج معقدة للإجابة حتى عن أبسط الاستفسارات التى تجرى على قاعدة البيانات.

٢- محدودية الاستقلالية بين البيانات والبرامج التى تتعامل معها، ومن ثم فإن البرامج ليست بمعزل عن «هيئة البيانات» (Data Format).

٣- لا يوجد لأى من النموذجين أسس نظرية مقبولة بشكل كبير تمكن من فهم وتحليل أثــر التعامل مع البيانات المخزنة فى قاعدة البيانات أو نتائج الاستفســارات التى تطبق على قاعدة البيانات.

وقد حدت المســاوئ التى تشوب كلا النموذجين السابقين بأحد الباحثين فى شركة أي. بي. إم. (IBM) ويدعى «إدغار كود» (Edgar Codd)، عام ١٩٧٠م. إلى اقتراح نموذج جديد ســمى بالنموذج العلاقى (Codd, 1970). ويمثل النمــوذج العلاقى الجيل الثانى لنظم إدارة قواعد البيانات.

الثمانينيات الميلادية:

لاقــى النمــوذج العلاقى قبولاً كبيراً مــن المهتمين فى نظــم إدارة قواعد البيانات وانتشر استخدام هذا النموذج بشكل كبير على المستوى التجارى. واتسم هذا النموذج ببســاطته فى تنظيم البيانات، وبزيادة درجة عدم الاعتمادية (أو الارتباط) بين برامج التطبيقات من جانب والبيانات من جانب آخر. وتميز بســهولة الاســتخدام حتى من قبل غير المبرمجين، إذ إن كافة البيانات والعلاقات فيما بينها تمثل على هيئة جداول بســيطة يمكن فهم محتوياتها والتنقل بينها بســهولة كبيرة. كما صاحب هذا النموذج أســس نظرية تمكن من فهم وتحليل الطرق التى يتم التعامل فيها مع البيانات المخزنة فى قاعدة البيانات.

واستخدم مع هذا النموذج لغات تداول قواعد البيانات العلاقية (مثل SQL) تسمح بمعالجــة البيانات فى حالة مجموعات أو جداول (عوضاً عن الســجلات)، وبحيث لا يحتاج المســتفيد أو المبرمج وصف مســار البحث عن البيانات فى الاستفســارات أو برامــج التطبيقات كما هو الحال فى نمــاذج قواعد البيانات الأخرى. وتم تطوير لغة الاستفســار البنائية (SQL) من قبل شــركة أي. بي. إم. لتصبح جزءاً من مشروع نظام إدارة قاعــدة البيانــات المعروف بنظام «آر» (System R). كمــا تم وضع مقاييس لهذه اللغــة بنهاية الثمانينيات الميلادية. وتم تطوير هــذه المقاييس عدة مرات كان أخرها

عام ١٩٩٩م، وتبنى هذه المقاييس معهد المقاييس الوطنى الأمريكى (American National Standards Institute (ANSI) والمنظمــة الدولية للمقاييس (International Organization for Standardization (ISO).

وبناءً على مجهودات «كود» فى مجال نظم قواعد البيانات: حاز على جائزة تورينغ عام ١٩٨١م.

التسعينيات الميلادية:

تطورت عمارة الحاسبات الآلية (Computer Architecture) ونظم الاتصالات والشبكات تطــوراً كبيراً خلال هذا العقد. ومع هذا التطــور تطورت طبيعة التطبيقات وظهرت مفاهيــم حديثــة لم تكن معروفة فيما ســبق إلا كضرب من الخيــال العلمى المحدود جــداً إذا ما قورن بكمية ونوعيــة التطبيقات والمفاهيم التى ظهــرت فى هذا العقد. فمع تطور نظم الحاســبات الآلية ونظم الاتصالات والشبكات أصبح بالإمكان معالجة ونقل كميات كبيرة من البيانات، مثل الرسومات، والصور، ولقطات الفيديو، والصوت. وعليه أصبح بالإمكان تطويــر تطبيقات جديدة تلبى احتياجات المفاهيم الحديثة مثل التجارة الإلكترونية، والتعليم عن بعد، والحكومة الإلكترونية، والحرب الإلكترونية، على ســبيل المثال لا الحصر. ونتيجة للكم الهائــل من البيانات التى تتعامل معها مثل هذه التطبيقات كان لزاماً أن تتطور نظم إدارة قواعد البيانات. وبالفعل تطورت هذه النظم وأخذت عملية التطوير شكلين رئيسيين: الأول منهما تمثل فى نموذج قواعد البيانات الشــيئى (Object-Oriented Data Model) الذى بدأ ظهوره الفعلى فى أواخر الثمانينيات الميلاديــة. وظهر العديد من نظم إدارة قواعد البيانات المبنية على هذا النموذج. أما التطور الثانى فتمثل فــى تحديث بعض منتجى نظم إدارة قواعد البيانات المبنية على النموذج العلاقى لنظمهم بحيث تحتوى على بعض مفاهيم النموذج الشيئى. وأصبحت هذه النظم معروفة بقواعد البيانات العلاقية الشيئية (Object-Relational Databases).

بداية القرن الحادى والعشرين وما بعد:

دخلت نظم إدارة قواعد البيانات فى منظومة شــبكة الإنترنت، فأصبح الكثير من مواقــع الإنترنت يعتمد فى تخزين وإدارة بياناتها على نظم قواعد البيانات عوضاً عن الملفات التقليدية التى كانت تعتمد عليها مواقع الإنترنت فى تخزين بياناتها عند بداية ظهورها. وأصبح بالإمكان تطوير نماذج لصفحات الإنترنت يتم اســتخدامها من قبل

متصفحات الإنترنت لكتابة الاستفســارات (Queries) ومن ثم تنفيذ هذه الاستفسارات علــى قاعدة بيانات الموقــع. وبعد الحصول على نتيجة الاستفســارات تهيئ النتيجة باســتخدام إحدى لغــات المتصفحــات مثــل Hyper Text Mark-up Language (HTML) لعرضها من خلال المتصفح (Browser).

ومع دخول نظم إدارة قواعد البيانات فى منظومة شبكة الإنترنت، فإن ذلك أعطاها زخماً جديداً من الأهمية وضرورة البحث عن طرق وأســاليب جديدة للاستخدام فى تصميمها بهــدف تلبية الاحتياجات الجديــدة التى تتطلبها تطبيقــات الإنترنت مثل الوسائط المتعددة من صورة وصوت.

الفصل الثانى

نمذجة بيانات المنظمة

يعـد «تجريد البيانـات» (Data Abstraction) إحدى الخصائص الأساسـية لقواعد البيانـات، إذ يمكّن من إخفاء تفاصيل حفظ البيانات عن المسـتخدمين، ومن ثم فإنه يعفيهـم من الخوض فـى هذه التفاصيل عنـد تداولهم للبيانـات المخزنة فى قاعدة البيانـات. ويمكـن نموذج البيانات، الذى يعرف عادة علـى أنه مجموعة من المفاهيم التى تمكن من وصف تركيبة (Structure) مكونات قاعدة البيانات (Elmasri and Navathe, 2004)، من الوصول إلى هذا المسـتوى من التجريـد. ويقصد بتركيبة قاعدة البيانات «نوعية البيانات» (Data Types)، والعلاقات فيما بينها، والقيود المفروضة عليها، ويجب أن تتحقـق على أية حالة من الحالات التى قد تكون عليها قاعدة البيانات. كما توفر معظـم نماذج البيانات بعض العمليات الأساسـية لتداول البيانات من اسـترجاع لها وتحديث عليها.

نموذج البيانات هو مجموعة من المفاهيم التى تمكن من وصف تركيبة (Structure) مكونات قاعدة البيانات.

ويمكن تصنيف نماذج البيانات وفق نوعية المفاهيم التى تستخدمها لوصف تركيبة قاعدة البيانات. فالنمذجة عالية المسـتوى أو المفاهيميـة للبيانات (Conceptual Data Modeling) توفر مفاهيم قريبة من إدراك المسـتفيدين للبيانات، فى حين أن النمذجة متدنية المسـتوى أو المادية للبيانـات (Physical Data Modeling) فتقدم مفاهيم لوصف تفاصيل تخزين البيانات على الحاسب الآلى، وهى موجهة بشكل عام إلى المتخصصين فى الحاسـب الآلى وليس إلى المسـتفيدين منه. وبين هاتين النهايتين يوجد نوع ثالث يسـمى النمذجـة «التمثيليـة» (Representational) أو «التطبيقيـة» (Implementation) للبيانات. وهذا النوع من نماذج البيانات يوفر مفاهيم يمكن فهمها من قبل المستفيدين الذيـن لا يبعدون كثيراً عن الحاسـب الآلـى وطريقة تنظيم البيانـات عليه. كما أن النمذجـة التمثيلية للبيانات تخفى بعض تفاصيل تخزين البيانات وفى الوقت نفسـه يمكن استخدامها بشكل مباشر على الحاسب الآلى.

تستخدم النمذجة المفاهيمية مفاهيم، مثل الكينونة، والخاصية، والعلاقة. وتمثل الكينونة شيئاً حقيقياً موجوداً على أرض الواقع أو مفهوماً معيناً. فمن الأشياء الموجودة على أرض الواقع الموظف، والطالب، والسيارة وما إلى ذلك من أشياء محسوسة. ومن أمثلة المفاهيم الحساب البنكى، والمشروع، والقسم الدراسى أو الإدارة، وما إلى ذلك من أشياء غير محسوسة ولكنها ذات معنى فى بيئة المستفيد من قاعدة البيانات. أما الخاصية فهى سمة تصف الشىء مثل اسم الموظف أو مرتبته الوظيفية أو راتبه الشهرى، فى حين أن العلاقة ارتباط بين كينونتين أو أكثر. فعلى سبيل المثال توجد علاقة بين الموظف والإدارة التى يعمل فيها وتسمى مثل هذه العلاقة علاقة «يعمل فى». كذلك هو الحال بالنسبة لعلاقة الطالب بالقسم الدراسى أو الكلية الجامعية، وتسمى «يدرس فى». ويتطرق هذا الفصل لأحد النماذج المفاهيمية ويسمى «نموذج كينونة – علاقة» (Entity-Relationship Model) الذى يعد أكثر النماذج المفاهيمية عالية المستوى شيوعاً. أما الفصل التالى فيقدم مفاهيم إضافية تستخدم فى النموذج المفاهيمى كينونة – علاقة مثل «التعميم» (Generalization) و«التخصيص» (Specialization).

أما النمذجة التمثيلية للبيانات أو التطبيقية فهى أسلوب النمذجة المستخدم فى نظم إدارة قواعد البيانات على المستوى التجارى. ومن هذه النماذج «النموذج العلاقى» (Relational Model) الذى يعد الأوسع انتشاراً فى وقتنا الراهن، و«النموذج الشبكى» (Network Model) و«النموذج الهرمى» (Hierarchical Model) اللذان كانا قيد الاستخدام وحتى وقت قريب.

أما النمذجة المادية للبيانات فهى نماذج تستخدم لوصف الكيفية التى يتم فيها تخزين البيانات فى ملفات على الحاسب الآلى من خلال توفيرها لطرق من شأنها تهيئة السجلات (أو تشكيلها) (Record Formatting) فى الملفات، وترتيب السجلات داخل الملفات (Record Orderings)، والوصول إلى السجلات (Access Paths). أما طرق الوصول إلى السجلات فما هى إلا هياكل (Structures) من شأنها أن تعجل أو تسرع فى عملية البحث والوصول إلى سجلات البيانات المخزنة فى الملفات.

٢-١ نمذجة البيانات وقواعد العمل باستخدام النمذجة المفاهيمية (Conceptual Modeling):

يتم التعرف على قواعد العمل (Business Rules) من خلال ما تَتّبعه أو تقوم به أية منظمة من سياسات، وإجراءات، وأحداث، ووظائف (Functions)، وأية أمور أخرى

تتعلق بطبيعة عمل المنظمة أو تقيدها. وتعد قواعد العمل ذات أهمية كبيرة فى منظومة نمذجة البيانات؛ لأنها توضح كيفية تداول البيانات وتخزينها. وتعتبر أسماء البيانات والتعاريف أبسط أنواع قواعد العمل، ففى النمذجة المفاهيمية للبيانات يجب تسمية وتعريف الكينونات، وخصائصها، والعلاقات فيما بينها. ومن قواعد العمل الأخرى ما يمكن أن يضع بعض القيود على البيانات بحيث يمكن أن تمثل هذه القيود من خلال النموذج المفاهيمى.

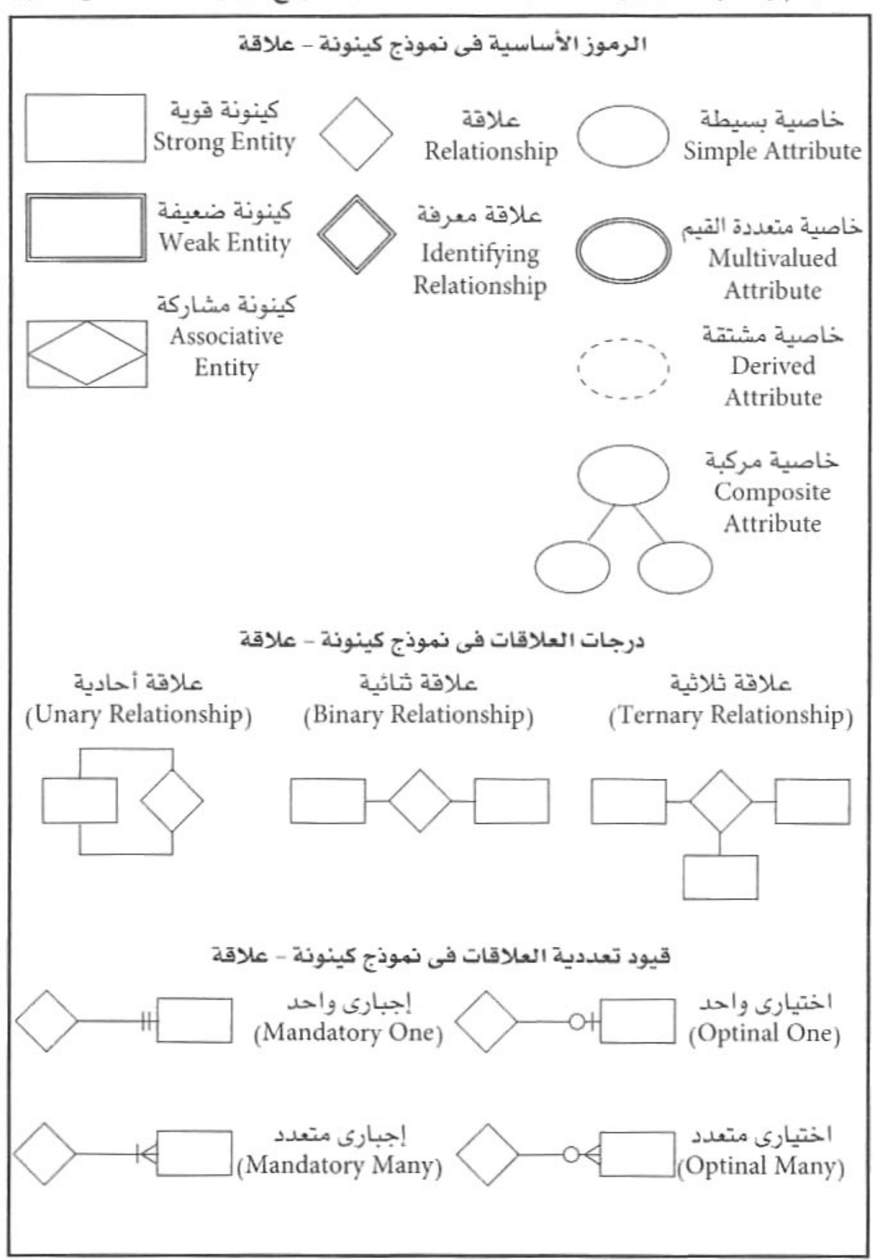
ويعد نموذج البيانات «كينونة - علاقة» (Entity-Relationship Model) أكثر نماذج البيانات المفاهيمية شيوعاً بسبب عدة عوامل من ضمنها السهولة النسبية فى الاستخدام، وإمكانية نمذجة البيانات وفق هذا النموذج باستخدام ما توفره غالبية «أدوات هندسة البرمجيات» (CASE Tools) من أدوات مخصصة للنمذجة، هذا بالإضافة إلى الاعتقاد السائد بأن الكينونات والعلاقات هى مفاهيم نمذجة لها القدرة على تمثيل الأشياء بشكل أقرب ما يكون من وجودها فى الطبيعة.

ونموذج البيانات كينونة - علاقة فى كثير من الأحيان هو أداة للتواصل بين الفنيين من مصممى قواعد البيانات والمستفيدين النهائيين من النظم، وذلك خلال فترة تحليل النظام. كما يستخدم نموذج كينونة - علاقة لبناء نموذج بيانات مفاهيمى يمثل تركيبة قاعدة البيانات والقيود عليها بمعزل عن البرمجيات التى ستستخدم لبناء قاعدة البيانات، متضمناً ذلك نظام إدارة قاعدة البيانات ونموذج البيانات التمثيلى أو التنفيذى الذى سيستخدم لبنائها. وتعد النمذجة المفاهيمية لقاعدة البيانات مرحلة ذات أهمية كبيرة فى تطوير «نظم تطبيقية» (Application Programs) ناجحة. لذلك فإن مصممى قواعد البيانات يأخذون الوقت الكافى فى هذه المرحلة لمناقشة النموذج المفاهيمى مع المستفيدين للتأكد من أنه يعكس كافة بياناتهم والقيود عليها، وذلك قبل الانتقال إلى المرحلة التالية.

٢-١-١ مكونات نموذج البيانات كينونة - علاقة:

نموذج البيانات كينونة - علاقة هو تمثيل منطقى مفصل للبيانات الموجودة فى المنشأة أو «منطقة العمل» (Business Area). ويمثل نموذج البيانات كينونة - علاقة من خلال مجموعة من الكينونات، وخصائصها، والعلاقات فيما بينها. ويوضح النموذج من خلال مخطط كينونة - علاقة الذى يتم تمثيله من خلال مجموعة من الرسومات. ويبين الشكل رقم (٢-١) أشكال الرموز الأساسية المستخدمة فى نموذج كينونة - علاقة ومعانيها.

شكل رقم (٢-١): الرموز الأساسية المستخدمة فى نموذج كينونة - علاقة ومعانيها



ولشــرح مكونات نموذج البيانات كينونة - علاقة سنســتخدم مثالاً لأحد تطبيقات قواعد البيانات المســتمدة من نظام لتســجيل الطلبة فى إحــدى الجامعات الأهلية. وســنفترض هنا أنه بعد عملية جمــع متطلبات النظام وتحليلها أثنــاء مرحلة تحليل النظام، تم استخلاص قواعد العمل التالية من قبل مصممى قواعد البيانات:

قواعد العمل المعمول بها فى الجامعة:

تنفــذ الجامعة الأهلية مجموعة من المواد الدراســية فى كل فصل دراســي. ومن قواعد العمل المتبعة فى الجامعة الأهلية ما يلى:

١- يوجد فى الجامعة عدد من الأقسام الدراسية، ولكل قسم (Department) من الأقسام العلمية رمز (Department_ID) يميزه عن بقية الأقسام، واسم (Name).

٢- يعمــل فــى الجامعة عدد مــن أعضاء هيئــة التدريس، ولكل عضــو هيئة تدريس (Faculty)[*] رقم (Faculty_ID) يميزه عن بقية أعضاء هيئة التدريس، واســم (Name) يتكون من (الاســم الأول (FName) واســم العائلة (LName))، وراتب شهرى (Salary)، وتاريخ ميلاد ((Date of Birth (DOB))، ورقم هاتف (Phone_No).

٣- يدرس فى الجامعة عدد من الطلاب، ولكل طالب (Student) رقم (Student_ID) يميزه عــن بقية الطلاب فى الجامعة، واســم (Name) يتكون من (الاســم الأول (FName) واسم العائلة (LName))، وعنوان بريدى (Address) يتكون من (اسم الشارع (Street)، واسم المدينة (City)، والرمز البريدى (Zip_Code)).

٤- تنفــذ الجامعــة مجموعة من المواد الدراســية، ولكل مادة دراســية (Course) رمز (Course_ID) يميزها عن بقية المواد الدراســية التى تنفذها الجامعة، واسم (Title)، وعدد وحدات (أو ساعات) دراسية (Units).

٥- تنفذ (أو تعقد) كل مادة دراســية من خلال مجموعة (أو شــعبة) دراسية (Section) واحدة أو أكثر فى الفصل الدراســى الواحد. أو قد لا تنفذ (أو تعقد) أية مجموعة (أو شــعبة) للمادة الدراسية فى فصل دراســى معين، ولكل مجموعة دراسية رمز

(*) تســتخدم كلمــة (Faculty) لتعنى «كليــة» أو كافة التابعين لها من طلبــة وموظفين على اختلاف طبيعة أعمالهم، إلا أن هذه الكلمة تستخدم أيضاً فى شمال أمريكا (الولايات المتحدة الأمريكية وكندا) للدلالة على الموظفين فى حقل التعليم وخاصة الجامعات والكليات العلمية الذين يقومون بالتدريــس دون ســواهم من الموظفين الذين لا يقومون بمهــام التدريس. وفى هذه الحالة تكون الكلمة مكافئة لكلمة «أستاذ» (Professor) أو محاضر (Lecturer).

(Section_ID) يتكون من (رقم المجموعة، والفصل الدراسى المنفذة فيه (Semester)، والسنة الدراسية المنفذة فيها (Year)). أما رقم المجموعة (Section_No) فهو رقم (مثل ١، ٢، ٣، ... إلخ) يميز المجموعة عن بقية المجموعات المنفذة للمادة الدراسية نفسها وفى نفس الفصل والسنة الدراسيين ولكنه لا يميزها بشكل منفرد عن بقية المجموعات الدراسية المنفذة للمواد الدراسية الأخرى فى الجامعة.

٦- قد يكون للمادة الدراسية الواحدة مجموعة من المتطلبات الدراسية، أو قد لا يكون للمادة الدراسية أية متطلبات دراسية. كما أن المادة الدراسية الواحدة قد تكون متطلباً لأكثر من مادة دراسية أو قد لا تكون متطلباً لأية مادة دراسية.

٧- يعمل (works for) فى كل قسم من أقسام الجامعة عضو هيئة تدريس واحد أو أكثر، وكل عضو من أعضاء هيئة التدريس يعمل فى قسم دراسى واحد فقط.

٨- كل عضو هيئة تدريس فى الجامعة مؤهل (Qualified) لتدريس مادة دراسية واحدة على الأقل، وقد يتوافر للمادة الدراسية الواحدة أكثر من عضو هيئة تدريس مؤهلاً لتدريسها أو قد لا يوجد من أعضاء هيئة التدريس من هو مؤهل لتدريس المادة.

٩- عندما يتأهل عضو هيئة التدريس لتدريس مادة ما لأول مرة، يكون هنالك تاريخ لتأهيله (Qualification date) يحدد تاريخ تأهل عضو هيئة التدريس لتدريس المادة الدراسية.

١٠- تدار (Administered) كل مادة دراسية من قبل قسم دراسى واحد من أقسام الجامعة، ويدير كل قسم مادة دراسية واحدة على الأقل.

١١- قد يسجل (Enrolls) الطالب الواحد فى أكثر من مجموعة (أو شعبة) دراسية أو قد لا يسجل فى أية مجموعة (أو شعبة) دراسية، والمجموعة (أو الشعبة) الدراسية الواحدة قد لا يسجل فيها أى طالب أو قد يسجل فيها أكثر من طالب.

١٢- عندما يسجل طالب فى مجموعة دراسية تكون له درجة (Grade) تعطى عند انتهائه من الدراسة فى المجموعة.

١٣- يتخصص كل طالب (Majors) فى قسم دراسى واحد فقط، ويتخصص فى القسم الدراسى الواحد أكثر من طالب.

١٤- يكلف (Assigned) كل عضو هيئة تدريس بتدريس مجموعة (أو شعبة) دراسية واحدة أو أكثر وقد لا يكلف عضو هيئة التدريس بأية مجموعة (أو شعبة) دراسية، والمجموعة (أو الشعبة) الدراسية الواحدة تكلف لعضو هيئة تدريس واحد فقط.

٢-١-٢ المكونات الأساسية لنموذج البيانات كينونة - علاقة:

يتكون نموذج البيانات المفاهيمى كينونة - علاقة من ثلاثة مكونات رئيسية هى:

- الكينونة.
- الخاصية.
- العلاقة.

وفيما يلى شرح لهذه المكونات الرئيسية.

٢-١-٢-١ الكينونة (Entity):

الكينونة هى شخص أو مكان أو شىء أو حدث أو مفهوم فى بيئة المنظمة ويراد الاحتفاظ ببيانات عنها.

ومن أمثلة كينونات الأشخاص كينونة الموظف وكينونة الطالب وكينونة العميل، أما كينونة المدينة، وكينونة السوق، وكينونة المبنى فتعتبر أمثلة لكينونات الأماكن. ومن أمثلة كينونات الأشياء كينونة سيارة، وكينونة منتج، وكينونة جهاز، أما كينونات المفاهيم فمن أمثلتها كينونة مادة دراسية، وكينونة رحلة جوية، وكينونة حساب بنكى. ويعنى هذا أنه ليس بالضرورة أن تمثل الكينونات أشياء ملموسة لها وجودها الفيزيائى فى الطبيعة، ولكنها قد تمثل أشياء أخرى لها مفهومها فى بيئة المنظمة.

٢-١-٢-١-١ الفرق بين فئة الكينونة وحالة من حالات الكينونة (Entity Type Versus Entity Instance):

يتم التفريق عادة بين فئة (أو نوع) الكينونة (Entity Class or Entity Type) عن حالات الكينونة (Entity Instances) حيث أن نوع أو فئة الكينونة يمكن أن يعرف كما يلى:

فئة الكينونة هى تمثيل لمجموعة من الحالات للأشياء التى لها خواص مشتركة.

فكينونة الطالب فى الواقع هى نوع أو فئة لمجموعة من حالات الطلبة، فالطالب أحمد يختلف باعتباره حالة عن الطالب صالح وعن الطالب عبدالعزيز، إلا أن جميعهم يشتركون

فـــى نفس الخواص، فكل منهم لديه اســـم أول، واســـم لعائلته، ورقم خـــاص به (كالرقم الجامعى للطالب على سبيل المثال)، كما أن كلاً منهم لديه هاتف وعنوان بريدى.

ويتـــم تمثيل الحالات للكينونات فى نموذج كينونـــة - علاقة من خلال تمثيل الفئة (أو النـــوع) التـــى تتبعها هذه الحالات دون تمثيل كل حالة بشـــكل منفرد فى النموذج، فمثلاً يتم تمثيل جميع الطلاب من خلال فئة الكينونة التى يتبعونها وهى كينونة طالب فـــى نموذج البيانات كينونة - علاقة عوضاً عـــن إدراجهم جميعاً فى النموذج. وبهذه الطريقة يمكن تمثيل بيانات أية منظمة من خلال تمثيل فئات الكينونات التى تحتويها. كما أن هذه الطريقة تعد مختصرة جداً مما يســـاعد على التعرف على بيانات المنشأة وتمثيلها بشكل مبسط.

وتمثل الكينونات فى نموذج كينونة - علاقة على شكل مستطيل يكتب بداخله اسم فئة (أو نوع) الكينونة. ومن الإرشـــادات التى تتبع عادة عند تســـمية الكينونات ما يلى :(Hoffer et al, 2002)

١- اســـم فئة الكينونة «مســـمى فردى» (Singular Noun) مثـــل «عميل» (CUSTOMER) وطالب (STUDENT). إلا أن مسمى فئة الكينونة قد يكون «مسمى جماعياً» (Plural Noun) مثل (CUSTOMERS) و(ACCOUNTS) ويكون هذا فى الحالة التى تكون فيها قراءة المخطط أفضل من استخدام مسمى فردى للكينونة.

٢- يجب أن يكون مســـمى الكينونة مخصصاً للمنظمة، فعلى سبيل المثال قد يستخدم المســـمى «طالب» (STUDENT) فى إحدى الجامعات أو المدارس ويستخدم المسمى «متدرب» (TRAINEE) فى المنظمات التى تقوم بالتدريب التطبيقى (أو العملى) عوضاً عن التدريس النظرى (أو الأكاديمى). كما يجب أن يكون المسمى ذا طبيعة وصفية تصف المقصود من الكينونة بشكل مختلف عن بقية الكينونات فى المنظمة.

٣- مســـمى فئة الكينونة يجب أن يكون محـــدداً بأقل قدر ممكن من الكلمات. فمثلاً يســـتخدم الاسم «تسجيل» (ENROLLMENT) لتمثيل فئة كينونة تسجيل الطلبة فى المواد الدراسية عوضاً عن استخدام «التسجيل فى مادة دراسية» (ENROLLMENT IN A CLASS). وعادة ما يفهم معنى المسمى من خلال علاقة فئة الكينونة بالفئات الأخرى فى المخطط.

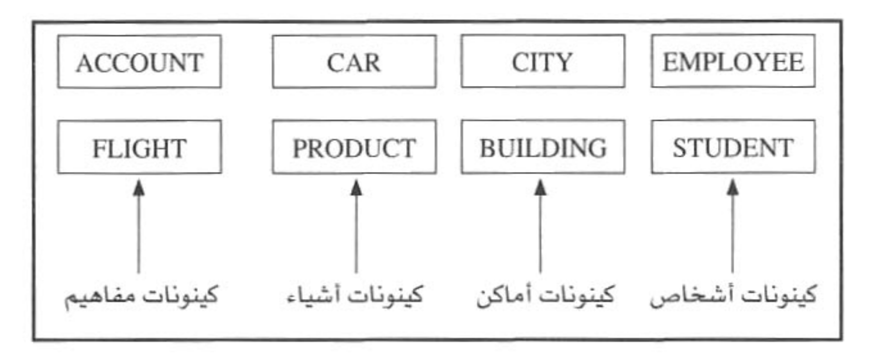
٤- فى حالة تمثيل الفئة لحدث معين فإن مســـمى الفئة يكون ممثلاً لنتيجة الحدث.

فعلى سـبيل المثال عند تسـمية حدث تسجيل الطالب فى مادة معينة يكون مسمى الفئة (ENROLLMENT) وهو ممثل لنتيجة الحدث وهى عملية التسجيل.

٥- عند اسـتخدام اسـم معين لفئة كينونة فإنه يجب استخدام نفس المسمى فى كافة مخططات كينونة - علاقة الخاصة بالمنظمة.

ويوضح الشـكل رقم (٢-٢) بعـض الأمثلة لفئات الكينونـات وطريقة تمثيلها فى مخطط كينونة علاقة.

شكل رقم (٢-٢): أمثلة لفئات الكينونات وطريقة تمثيلها فى مخطط كينونة - علاقة



٢-١-٢-١-٢ خصائص الكينونات (Entity Attributes):

ترتبـط كل فئة كينونة بعدد من الخصائـص (Attributes). والخاصية هى صفة أو سـمة لفئة الكينونة التـى يراد تمثيل بياناتها فى قاعـدة البيانات الخاصة بالمنظمة. فالاسـم الأول والاسـم الأوسط واسـم العائلة قد تكون بعض خصائص كينونة عميل (CUSTOMER)، ونوع السـيارة ورقم لوحتها والبلـد الذى صنعت فيها قد تكون بعض خصائـص كينونة مركبة قيادة (VEHICLE). ويمكن تصنيف خصائص الكينونات إلى أربعة أنواع رئيسية وهى:

١- **الخاصية البسيطة (Simple Attribute):** هى الخاصية التى لا يمكن أن تنقسم إلى خصائـص فرعية، وبحيث إنها لا يمكن أن تأخذ أكثـر من قيمة واحدة فقط مثل الاسـم الأول للموظف (Emploee_First_Name)، أو نـوع المركبة (Vehicle_Type)، أو رقم الموظف (Employee_Number). ويرمز للخاصية البسـيطة بالشكل البيضاوى مدوناً بداخله اسم الخاصية.

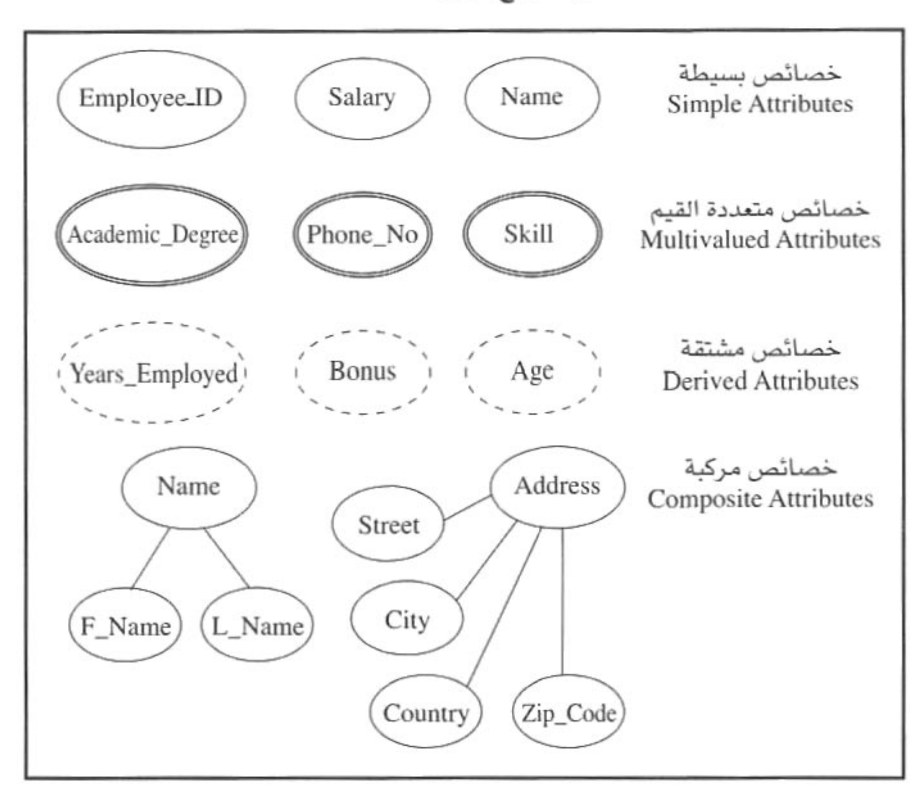
٢- **الخاصيــة المركبــة (Composite Attribute):** هـى الخاصية التى تتكون من مجموعة من الخصائص البســيطة مثل الاســم، حيث إنه قد يتكون من الاســم الأول واسم الأب واسم العائلة. كذلك هو الحال بالنسبة للعنوان البريدى الذى قد يتكون من: اسم الشــارع (Street)، واسم المدينة (City)، واسم البلد (Country)، ويرمز للخاصية المركبة بالشكل البيضاوى أيضاً ويكتب بداخلها اسم الخاصية، وبحيث يتفرع منها الخصائص البسيطة المكونة لها.

٣- **الخاصيــة متعــددة القيــم (Multivalued Attribute):** هـى الخاصية التى من الممكن أن تأخذ أكثر من قيمة مثل خاصية المهارة (Skill) للموظفين. فقد يكون للموظف، على ســبيل المثال، المهارة فى البرمجة بأكثر من لغة برمجة. أو قد يكون من المهم للمنظمة تدوين مهارات (أو قدرات) الموظفين من حيث اللغات التى يمكن التخاطب بهــا. كذلك هو الحال بالنســبة لأرقام الهواتف والدرجــات العلمية عندما تتعدد عند الموظفين، على ســبيل المثال، وترغــب المنظمة فى تمثيل بياناتها ضمن قاعدة البيانات. ويرمز للخاصية متعددة القيم بالشكل البيضاوى المزدوج الخطوط يدون بداخله اسم الخاصية.

٤- **الخاصية المشــتقة (Derived Attribute):** هى الخاصية التى يمكن اســتنتاجها من خلال خاصية (أو خصائص) أخرى للكينونة مثل العمر (Age) الذى يمكن حســابه بعملية طرح تاريخ اليوم (الذى يوفره نظام الحاسب الآلى) من خاصية تاريخ الميلاد التى تكون مصاحبة لفئة الكينونة. كذلك هو الحال بالنسبة لتاريخ تقاعد الموظف (Retirement_Date) الذى يعتبر خاصية مشتقة يمكن حسابها من خلال عملية جمع خاصية تاريخ ميلاد الموظف مع الســن التقاعدية المســموح بها فى أنظمة المنظمة أو الدولة. وكذلك هو الحال بالنســبة لخاصية المكافأة السنوية (Bonus) للموظف، فى بعض المنظمات، التى يمكن حسابها كنسبة من خاصية راتب الموظف (Salary). ويرمز للخاصية المشتقة فى نموذج كينونة - علاقة بالشكل البيضاوى المنقط يكتب بداخله اسم الخاصية.

وعند تسمية خصائص فئات الكينونات، على اختلاف أنواعها، يكون الحرف الأول من الخاصية حرفاً كبيراً (Capital Letter). وفى حالة كون اســم الخاصية مركباً فإنه يتــم الربط ما بــين الكلمات المكونة لاســم الخاصية بالعلامــة "__". كذلك يمكن اســتخدام الاختصــارات، ولكنها يجب أن تكون مفهومة ومدونة بشــكل واضح ضمن وثائق النظام. ويحتوى الشــكل رقم (٢-٣) على بعض الأمثلة التى توضح طرق تمثيل الأنواع الأربعة من الخصائص.

شكل رقم (٢-٣): أمثلة توضح طرق تمثيل الأنواع الأربعة من الخصائص فى نموذج كينونة - علاقة



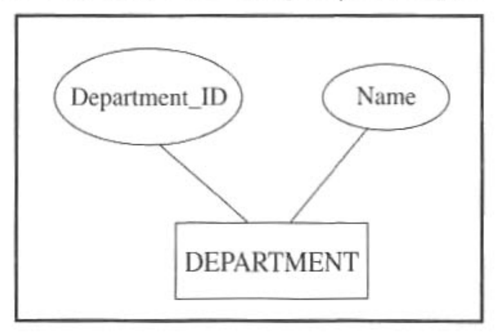
وتربط كل خاصية بفئة الكينونة التابعة لها بخط مستقيم. فعلى سبيل المثال لنأخذ قاعدة العمل الأولى والثانية فى الجامعة الأهلية، وقد سبق ذكرهما فى هذا الفصل ونحاول تمثيلها فى مخطط كينونة - علاقة.

قاعدة العمل (١): يوجد فى الجامعة عدد من الأقسام الدراسية، ولكل قسم (Department) من الأقسام العلمية رمز (Department_ID) يميزه عن بقية الأقسام، واسم (Name).

نظراً لوجود عدة حالات للقسم الدراسى فمنها، على سبيل المثال، قسم الحاسب الآلى وقسم الرياضيات، ... إلخ، وهو ما نصت عليه قاعدة العمل؛ تمثل قاعدة العمل

هذه فى مخطط كينونة - علاقة كفئة كينونة بمسمى (DEPARTMENT) ولها خاصيتان بسيطتان هما رمز القسم (Department_ID) واسم القسم (Name) كما هو موضح فى الشكل رقم (٢-٤).

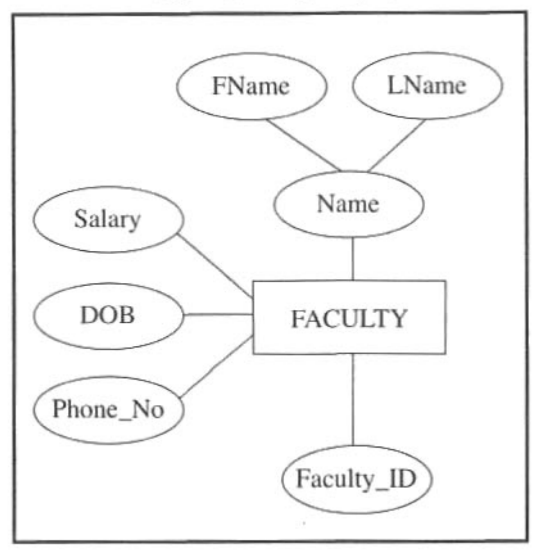
شكل رقم (٢-٤): تمثيل القسم الدراسى كفئة كينونة ذات خاصيتين بسيطتين



قاعدة العمل (٢): يعمل فى الجامعة عدد من أعضاء هيئة التدريس، ولكل عضو هيئة تدريس (Faculty) رقم (Faculty_ID) يميزه عن بقية أعضاء هيئة التدريس، واسم (Name) يتكون من (الاسم الأول (FName) واسم العائلة (LName))، وراتب شهرى (Salary)، وتاريخ ميلاد ((Date of Birth (DOB)، ورقم هاتف (Phone_No).

نظراً لوجود العديد من حالات أعضاء هيئة التدريس فى الجامعة، تمثل قاعدة العمل هذه فى مخطط كينونة - علاقة كفئة كينونة بمسمى (FACULTY) ولها أربع خصائص بسيطة هى: رقم عضو هيئة التدريس (Faculty_ID) وتاريخ ميلاده (DOB)، وراتبه الشهرى (Salary)، ورقم هاتفه (Phone_No)، وخاصية مركبة هى اسمه (Name) التى تتفرع إلى خاصيتين بسيطتين هما الاسم الأول (FName)، واسم العائلة (LName) وذلك كما هو موضح فى الشكل رقم (٢-٥).

شكل رقم (٢-٥): تمثيل عضو هيئة التدريس كفئة كينونة ذات أربع خصائص بسيطة وخاصية مركبة



٢-١-٢-١-٣ الخاصية المميزة (Identifying Attribute) لفئة الكينونة:

الخاصية المميزة لفئة الكينونة هى واحدة أو أكثر من خصائص فئة الكينونة بحيث تحدد هذه الخاصية بشكل منفرد كل حالة من حالات الكينونة، وفى الوقت نفسه لا تتغير بتغير الزمن. وبمعنى آخر تستخدم الخاصية المميزة للتفريق ما بين الحالات التى تمثلها فئة الكينونة. فعلى سبيل المثال، الخاصية المميزة لكينونة القسم الدراسى فى الجامعة الأهلية هى رمز القسم؛ وذلك لأن رمز القسم يختلف باختلاف الأقسام العلمية ودون تكرار بحيث لا يمكن أن يكون لقسمين دراسيين مختلفين نفس الرمز. كما أن رمز القسم لا يتغير بتغير الزمن. أما بالنسبة لفئة كينونة أعضاء هيئة التدريس (FACULTY) فإن الخاصية المميزة لفئة الكينونه هى رقم عضو هيئة التدريس (Faculty_ID)؛ لأنه يميز كل عضو هيئة تدريس عن بقية أعضاء هيئة التدريس ولا يتغير بتغير الزمن. أما بالنسبة لبقية الخصائص فى كينونة أعضاء هيئة التدريس مثل تاريخ الميلاد ورقم الهاتف فإنها لا تصلح لأن تكون مميزاً لفئة الكينونة. فتاريخ الميلاد قد يتكرر بين الموظفين ومن ثم لا يمكن استخدامه للتفريق بين حالات الكينونة. أما رقم الهاتف، لو افترضنا أنه لا يتكرر بمعنى عدم اشتراك أكثر من عضو هيئة

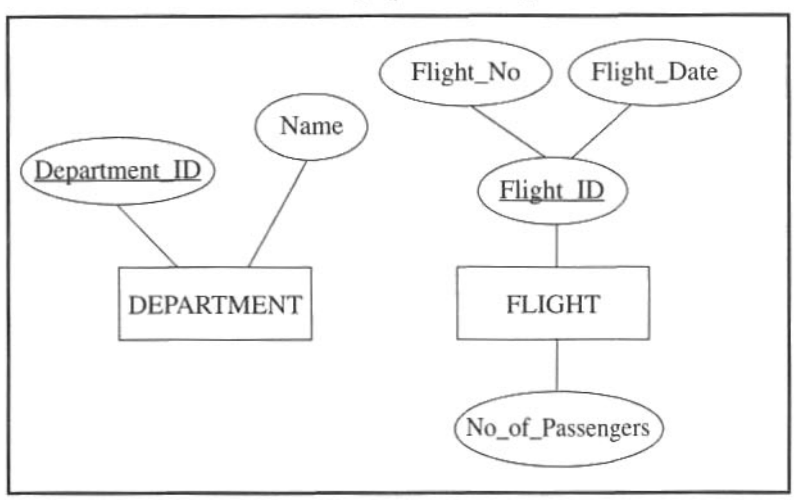
تدريــس لنفس رقــم الهاتف، فإنه لا يصلح أيضاً لأن يكــون خاصية مميزة لأنه يتغير بتغير الزمن، وذلك عند انتقال عضو هيئة التدريس من قســم إلى قسم أو مكتب آخر داخل الجامعة.

فى بعض الأحيان قد لا يوجد لفئة كينونة ما خاصية واحدة تصلح أن تكون خاصية مميــزة. فى هذه الحالة يتم اختيار أكثر من خاصية بحيث تمثل فى مجملها خاصية مميزة للكينونة بإمكانها التفريق بين حالات فئة الكينونة بشــكل منفرد. فعلى ســبيل المثال، وبافتراض عدم وجود خاصية رقم عضو هيئة التدريس وأن أسماء أعضاء هيئة التدريس لا تتكرر، يمكن اســتخدام اســم عضو هيئة التدريس الذى يتكون من الاسم الأول واسم العائلة كخاصية مميزة لفئة «كينونة أعضاء هيئة التدريس» (FACULTY). وفى هذه الحالة تصبح الخاصية المميزة خاصية مميزة مركبة (Composite Identifying Attribute). ولكن لو افترضنا أن أســماء أعضاء هيئــة التدريس تتكرر فإنه بالإمكان استخدام اســم عضو هيئة التدريس وتاريخ ميلاده كخاصية مميزة مركبة عوضاً عن الخاصية المركبة المتمثلة فى اسم عضو هيئة التدريس فقط. وكمثال آخر للخصائص المركبــة، لنفترض وجود الكينونــة رحلة طيران (FLIGHT) التــى لديها الخاصية رمز الرحلــة (Flight_ID) الذى يتكون من رقم الرحلــة (Flight_No) وتاريخ الرحلة (Flight_Date)، والخاصية عدد الركاب (No_of_Passengers).

يتــم اختيار الخاصية رمز الرحلة المكونة من رقــم الرحلة وتاريخ الرحلة كخاصية مميزة؛ إذ إنه لا يمكن استخدام أى من الخاصية رقم الرحلة أو تاريخ الرحلة كل على حدة كمميز. فرقم الرحلة يتكرر ولكن بتواريخ مختلفة، ومن ثم فإنه لا يميز كل رحلة على حدة بشــكل منفرد. كذلك هو الحال بالنســبة لتاريخ الرحلة، لأنه من الممكن أن تكون هنالك أكثر من رحلة فى التاريخ نفســه، ومــن ثم لا تصلح هذه الخاصية أيضاً لتصبح مميزاً لفئة الكينونة لأنها قد تتكرر لأكثر من رحلة.

وللتفريــق بين الخاصية المميزة وبقية خصائص فئة الكينونة فى مخطط كينونة - علاقة، يكتب اســم الخاصيــة المميزة وتحتها خــط (Underlined) كما هو موضح فى الشــكل رقم (٢-٦) الــذى يحتوى على مثالــين: أحدهما لفئة كينونــة الرحلة ذات الخاصية المميزة المركبة؛ وثانيهما لفئة كينونة القســم الدراسى ذات الخاصية المميزة البسيطة.

شكل رقم (٢-٦): التفريق بين تمثيل الخاصية المميزة وبقية خصائص الكينونة فى مخطط كينونة - علاقة



قد يوجد من ضمن خصائص فئة كينونة معينة أكثر من خاصية تصلح لأن تميز بين حالات الكينونة. فعلى سبيل المثال قد يكون لفئة كينونة الموظفين (EMPLOYEE) فى إحدى المنظمات خاصية تحتوى على رقم الموظف فى المنظمة (Employee_ID) وخاصية أخرى تحتوى على رقم السجل المدنى للموظف (Social_Identification_Number). فى هذه الحالة كلتا الخاصيتين تصلحان لأن تكونا خاصية مميزة لفئة الكينونة. وفى هذه الحالة يطلق على كل خاصية من الخاصيتين مسمى خاصية مميزة مرشحة (Candidate Identifying Attribute)، بمعنى أنه بالإمكان استخدام أى منهما مميزاً لفئة الكينونة. ولمصممى قاعدة البيانات الحرية فى اختيار إحدى الخصائص المميزة المرشحة لفئة الكينونة لتصبح الخاصية المميزة لفئة الكينونة. وتتم عملية الاختيار عند وجود أكثر من خاصية مرشحة وفق بعض المعايير التى بإمكان المصممين اتباعها ومنها ما يلى (Bruce, 1992):

١- أن لا تتغير قيمة المميز لكل حالة من حالات فئة الكينونة بتغير الزمن. فعلى سبيل المثال لا يحبذ استخدام الاسم الثلاثى حتى وإن ميز بين الموظفين فى المنظمة بشكل منفرد، فى بعض الدول، لأنه بإمكان الموظف أن يغير اسمه بشكل رسمى عند رغبته فى ذلك. ويعنى هذا أن المميز لهذا الموظف يجب أن يتغير من خلال تحديثه فى قاعدة البيانات عندما تحدث مثل هذه الحالة.

٢- أن تكون لكل حالة من حالات فئة الكينونة قيمة صحيحة للمميز، ولا يمكن أن تكون غيــر معرفة (Null) أو غير معلومة (Unknown). وفى حالة اختيار مميز مركب مثل (Flight_ID) فإنه يجب التأكد من أن كافة الخصائص المكونة للمميز سيكون لها قيم صحيحة ولا يمكن أن تكون غير معرفة أو غير معلومة.

٣- أن يكون المميز لفئة الكينونة هو المميز المرشح الأقل عدداً من الحقول.

٢-١-٢-١-٤ قواعد تسمية الخصائص:

هنالك بعض القواعد التى تتبع عادة عند تسمية الخصائص، بالإضافة إلى القواعد الرئيسية التى تتبع لتسمية الأشياء (أو الكينونات)، ومن هذه القواعد ما يلى:

١- يكون اسم الخاصية اسماً، مثل رقم الطالب (Student_Number)، والتخصص (Major)، وتاريخ الميلاد (Date_of_Birth)، ... إلخ. ولكون الخاصية مفهوماً (مثل رقم الرحلة (Flight_Number)) أو سمة فيزيائية (مثل الوزن (Weight)) للشىء قيد التمثيل؛ فإنه من الطبيعى أن توصف بأسماء لتمثيلها فى مخطط كينونة - علاقة.

٢- يجب أن يكون اسم الخاصية فريداً (Unique) ضمن أسماء خصائص فئة الكينونة، ويحبذ أن يكون فريداً أيضاً ضمن خصائص جميع الكينونات فى المخطط.

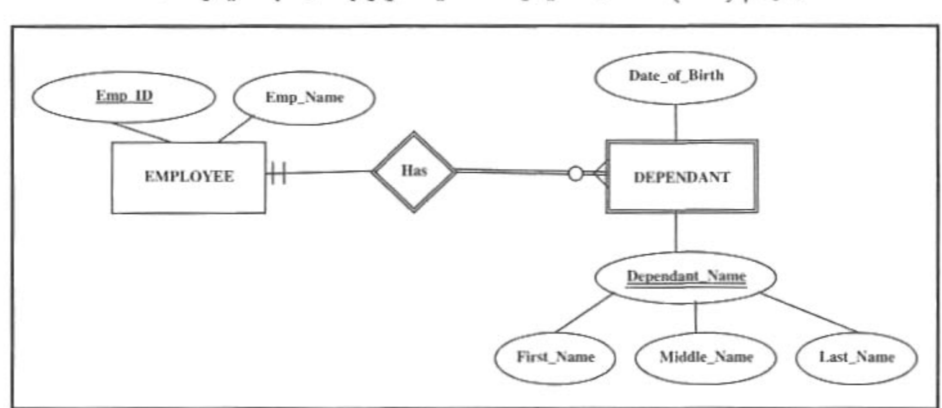
٢-١-٢-١-٥ الكينونة الضعيفة (Weak Entity):

إن غالبية فئات الكينونات التى يتم التعرف عليها فى أية منظمة هى فئات كينونات قويـــة (Strong Entities). وجميع الكينونات التى تطرقنـــا إليها حتى الآن هى من هذا النـــوع من فئات الكينونات. فالكينونة القوية توجد مســـتقلة عن بقية الكينونات ولها خاصيتها المميزة التى تمايز بين ما تمثله من حالات.

علـــى النقيض من ذلـــك، فإن الكينونـــة الضعيفة لا توجد مســـتقلة بل تعتمد فى وجودها علـــى فئة كينونة أخرى بدونها تصبح الكينونـــة الضعيفة غير ذات معنى فى مخطط كينونة - علاقة الذى يمثل قواعد عمل المنظمة. وتسمى الكينونة التى تعتمد عليهـــا الكينونة الضعيفة فى وجودهـــا «بالكينونة المُعرفة» أو «الكينونة المالكة». وعلى خلاف الكينونة القوية فإنه لا يوجد للكينونة الضعيفة خاصية مميزة تمايز بين حالاتها بشـــكل منفرد، بل يوجد فيها خاصية مميزة جزئية تمايز بين بعض حالاتها. ويحتوى مخطـــط كينونة - علاقة الموضح فى الشـــكل رقم (٢-٧) على مثـــال لكينونة ضعيفة

تمثل الأشـــخاص الذين يعولهـــم الموظف (DEPENDANT)، فى منظمـــة ما، والكينونة المالكـــة وهى كينونة الموظـــف (EMPLOYEE). وتعد كينونة الأشـــخاص الذين يعولهم الموظـــف كينونـــة ضعيفة؛ لأن هذه الكينونة لا وجود لها وغيـــر ذات معنى بدون وجود كينونة الموظف. فلو أخذنا أية حالة من حالات الكينونة الضعيفة، ولنقل محمد صالح عبداللـــه، فإن هذه الحالة لا وجود لهـــا؛ وإن وجدت فإنها غير ذات معنى ما لم يوجد صالح عبدالله (وهو والد أو معيل محمد) ضمن حالات كينونة الموظف المالكة للكينونة الضعيفـــة. ومعنى ذلك أن كل حالة من حـــالات الكينونة الضعيفة تعتمد فى وجودها على وجود حالة مقابلة لها فى الكينونة القوية.

شكل رقم (٢-٧): مثال لكينونة ضعيفة وارتباطها بالكينونة المالكة



وتمثل الكينونة الضعيفة فى مخطط كينونة - علاقة بمستطيل ذى حواف مزدوجة الخطـــوط، كمـــا تكتب الخاصية المميـــزة الجزئية وتحتها خطان كمـــا هو موضح فى خاصية اســـم الشـــخص الذى يعوله الموظف. وتميَّز العلاقة بـــين الكينونة الضعيفة والكينونة المالكة بالشـــكل المعين مـــزدوج الخطوط للدلالة علـــى أن هذه العلاقة هى العلاقة المعرفة التى بدونها لا توجد الكينونة الضعيفة. ويجب أن تكون درجة العلاقة واحـــد - متعدد بين الكينونة المالكـــة والكينونة الضعيفة، على التوالي. ويعنى هذا أن كل حالة من حالات الكينونة الضعيفة يجب أن ترتبط بحالة واحدة فقط فى الكينونة المالكة (هى التى أدت لوجودها أساســـاً)، فى حين قد ترتبط حالة من حالات الكينونة بأكثر من حالة فى الكينونة الضعيفة. فعلى ســـبيل المثال، يجب أن يرتبط كل شخص فى كينونة الأشـــخاص الذين يعولهم الموظفون بواحد فقط من الموظفين (هو الموظف

الذى يعوله)، فى حين قد يعول الموظف أكثر من شـــخص. أما التعددية الدنيا فتعتمد هنا على قاعدة العمل. ففى المثال السارى قد لا يعول أحد الموظفين أى شخص، وعليه فالتعدديـــة هنا اختيـــارى - متعدد كما هو موضح فى مخطـــط كينونة - علاقة. أما بالنســـبة للمميز الجزئى للكينونة الضعيفة الذى ســـبق أن أشرنا إليه على أنه للتمييز بـــين بعض حالات الكينونة الضعيفة، فهو فـــى الواقع للتمييز بين الحالات التابعة لكل حالة من حالات الكينونة المالكة. فعلى سبيل المثال، اسم الشخص الذى يعوله الموظف يعتبر مميزاً جزئياً؛ لأنه يميز بين الأشخاص الذين يعولهم الموظف الواحد، ولكنه ليس مميـــزاً كاملاً؛ لأنه لا يميز بين جميع حالات الكينونة الضعيفة؛ إذ إن بعض الأســـماء فيها قد تتكرر.

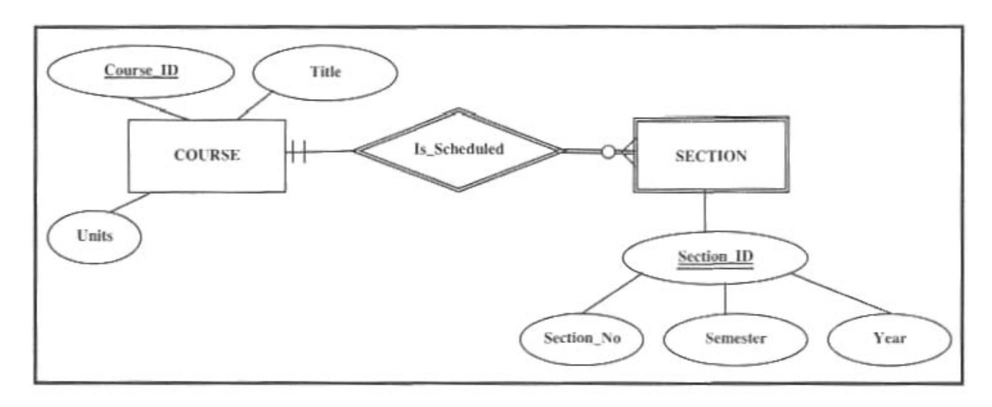
وكمثـــال آخـــر للكينونة الضعيفة، لنأخـــذ قاعدة العمل رقـــم (٥) من قواعد عمل الجامعة الأهلية ونضع لها النموذج المناسب فى مخطط كينونة - علاقة. تنص قاعدة العمل على التالى:

قاعدة العمل (٥): تنفذ (أو تعقد) كل مادة دراســـية من خلال مجموعة (أو شعبة) دراســـية (Section) واحـــدة أو أكثر فى الفصل الدراســـى الواحد، أو قـــد لا تنفذ (أو تعقد) أية مجموعة (أو شعبة) للمادة الدراسية فى فصل دراسى معين، ولكل مجموعة دراســـية رمز (Section_ID) يتكون من (رقم المجموعة، والفصل الدراســـى المنفذة فيه (Semester)، والســـنة الدراســـية المنفذة فيها (Year)). أما رقم المجموعة (Section_No) فهـــو رقم (مثل ١، ٢، ٣، ... إلخ) يميز المجموعة عـــن بقية المجموعات المنفذة للمادة الدراســـية نفسها (فى نفس الفصل والسنة الدراسيين)، ولكنه لا يميزها بشكل منفرد عن بقية المجموعات الدراسية المنفذة للمواد الدراسية الأخرى فى الجامعة.

يلاحظ فـــى قاعدة العمل الســـابقة أن المجموعة الدراســـية لا يوجد لها خاصية مميزة تميزها عن بقية المجموعات الدراســـية بشـــكل منفرد. فلو أخذنا، على سبيل المثال، مجموعة دراسية ما وافترضنا أن المميز للمجموعة الدراسية هو كل الخصائص البسيطة المكونة للخاصية المركبة رمز المجموعة وهى رقم المجموعة، والفصل الدراسى المنفذة فيه، والســـنة الدراســـية المنفذة فيها، فإن هذه الخصائص مجتمعة قد تتكرر لمجموعة دراســـية خاصة بمادة أخرى؛ لأن رقم المجموعـــة يميز بين مجموعات المادة الدراسية الواحدة المنفذة فى فصل دراسى من سنة دراسية ما، ولكنه لا يميز المجموعة عن مجموعة أخرى منفذة لمادة أخرى فى الفصل نفســـه من العام الدراســـى. ويعنى

هذا أننا لا نستطيع تمييز المجموعة الدراسية دون معرفة المادة الدراسية التى تتبعها المجموعة، ومن ثم فإن وجود أية مجموعة دراسية يعتمد على وجود المادة الدراسية التى تتبعها المجموعة. وبناءً على ذلك فإننا نمثل المجموعة الدراسية (Section) بوصفها كينونة ضعيفة وتكون العلاقة مع المادة الدراسية هى العلاقة المعرفة كما هو موضح فى الشكل رقم (٢-٨).

شكل رقم (٢-٨): تمثيل المجموعة الدراسية ككينونة ضعيفة وارتباطها بكينونة المادة الدراسية

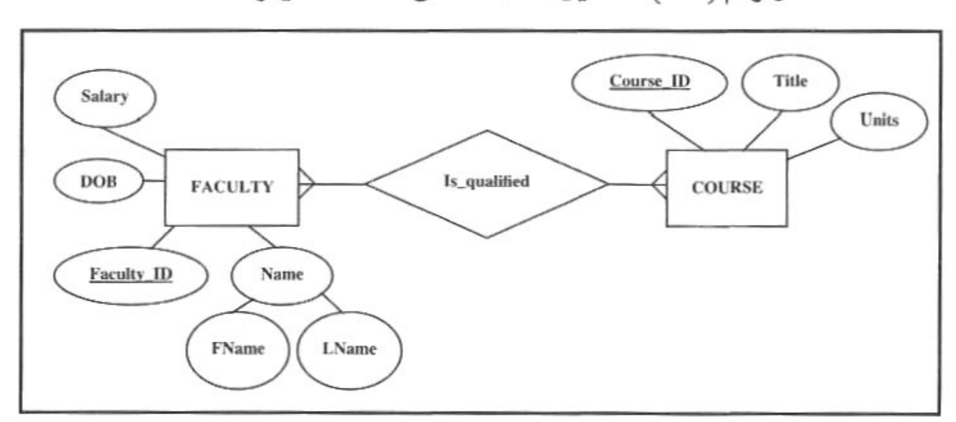


٢-١-٢-٢ العلاقات (Relationships):

العلاقة هى ارتباط بين حالات فئة كينونة ما بحالات فئة كينونة أخرى وذات أهمية للمنظمة، بحيث تسعى لتمثيلها ضمن مخطط كينونة - علاقة الذى ستشتق منه وتبنى قاعدة البيانات. ويعنى هذا أن العلاقات فى مخطط كينونة - علاقة هى الوسيلة التى تمكننا من الربط ما بين المكونات المختلفة للمخطط. وعادة يتم التفريق بين فئة العلاقة (Relationship Type) وحالات العلاقة (Relationship Instances)، كما هو الحال لفئة الكينونة وحالات الكينونة. ولإيضاح ذلك لنفترض فئة الكينونة عضو هيئة التدريس (FACULTY) وفئة الكينونة مادة دراسية (COURSE) مثل تلك الموجودة فى الجامعة الأهلية. ولنفترض أننا نرغب فى معرفة تأهيل كل عضو هيئة تدريس فى الجامعة للمواد التى بإمكانه تدريسها، بمعنى أننا نرغب فى معرفة كل المواد الدراسية المؤهل لتدريسها كل عضو هيئة تدريس فى الجامعة. فى هذه الحالة يتم تعريف فئة

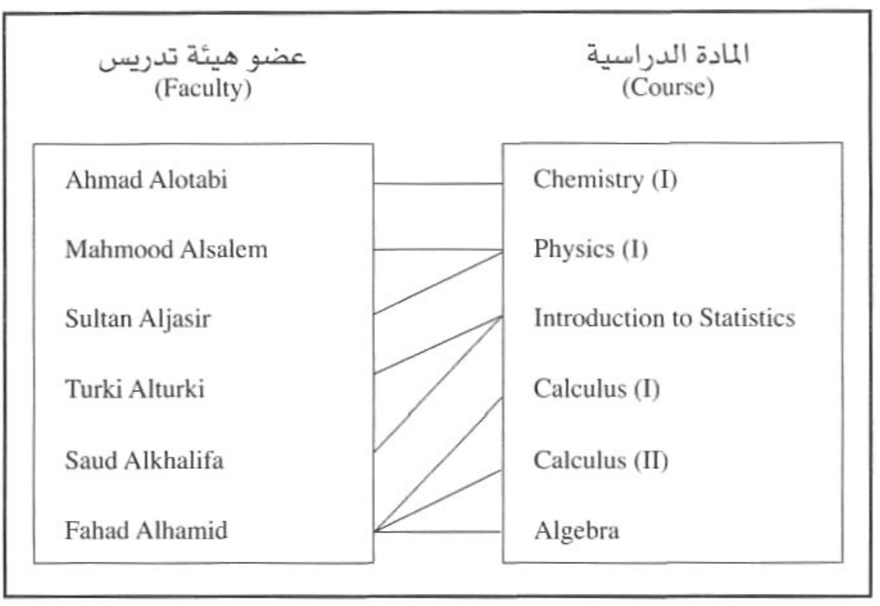
علاقة بمسمى «مؤهل لـ» (Is_qualified) بين فئة كينونة أعضاء هيئة التدريس وفئة كينونة المواد الدراسية. ولتمثيل علاقة ما يستخدم الشكل المعين مكتوباً بداخله اسم العلاقة، وبحيث يكون اسم العلاقة شبه جملة فعلية (Verb Phrase) كما هو موضح فى الشكل رقم (٢-٩). وتعد هذه العلاقة علاقة متعدد - متعدد، بمعنى أنه قد تكون لكل عضو هيئة تدريس القدرة على تدريس أكثر من مادة دراسية، كما أنه قد يكون للمادة الدراسية الواحدة أكثر من عضو هيئة تدريس مؤهل لتدريسها.

شكل رقم (٢-٩): تمثيل العلاقات فى مخطط كينونة - علاقة



ولإيضاح العلاقة السابقة فإن الشكل رقم (٢-١٠) يبين أن كلاً من محمود السالم وسلطان الجاسر مؤهلان لتدريس مادة الفيزياء (١) (Physics (I))، وأن المواد الثلاث حساب (١) وحساب (٢) والجبر (Calculus (I), Calculus (II) and Algebra) مؤهل لتدريسها عضو هيئة تدريس واحد هو فهد الحامد.

شكل رقم (٢-١٠): مثال لعلاقة أعضاء هيئة التدريس بالمواد الدراسية المؤهلين لتدريسها



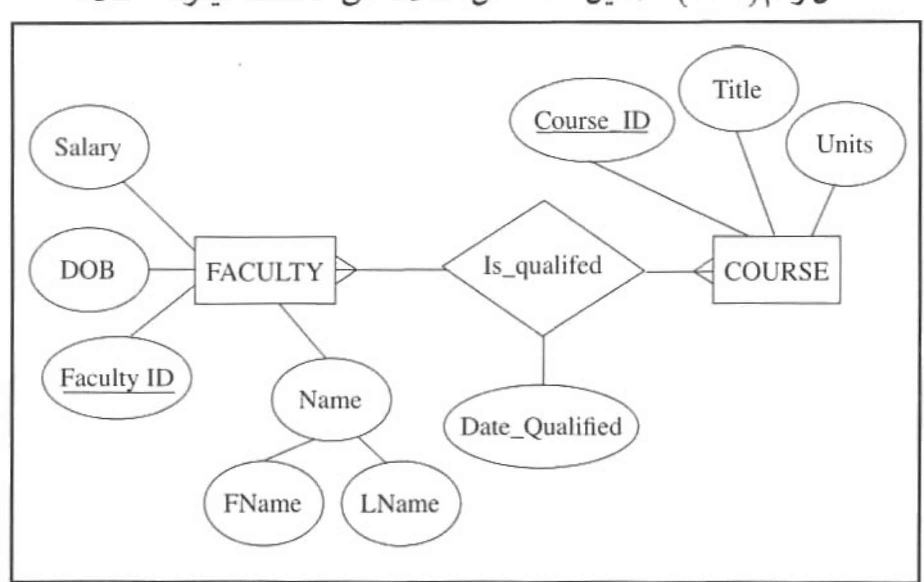
مما سبق يتضح أن فئة العلاقة ما هى إلا ارتباط ذو معنى بين فئتين من الكينونات (أو أكثر)، وأن فئات العلاقات تمكننا من ربط مكونات مخطط كينونة - علاقة مع بعضها بحيث نستطيع الإجابة عن تساؤلات لا يمكن الإجابة عنها باستخدام فئات الكينونات فقط. وتمثل فئة العلاقة بالشكل المعين يكتب بداخله اسم فئة العلاقة الذى يكون شبه جملة فعلية. كما أن أية حالة من حالات العلاقة تمثل ارتباطاً بين حالات الكينونات التى تربطها فئة العلاقة بحيث يوجد حالة واحدة فقط من كل فئة كينونة ترتبط بفئة العلاقة. فكل خط فى الشكل رقم (٢-١٠) يمثل حالة من حالات فئة العلاقة «مؤهل ل»، وهذه الحالة تمثل ارتباطاً بين حالة من فئة الكينونة «عضو هيئة التدريس» وحالة من حالات فئة الكينونة «مادة دراسية».

٢-١-٢-٢-١ خصائص العلاقة (Relationship Attributes):

قد يكون لفئة العلاقة خاصية أو أكثر كما هو الحال بالنسبة لفئات الكينونات. ففى حال رغبتنا، على سبيل المثال، فى تدوين التاريخ الذى تم فيه تأهل عضو هيئة التدريس لتدريس مادة معينة فإنه لا بد أن تكون خاصية تاريخ التأهل مرتبطة بفئة العلاقة وليس بأى فئة من الكينونتين اللتين تربط بينهما فئة العلاقة. والسبب وراء

ذلك هو أن تاريخ التأهل هو خاصية للعلاقة نفسها وليست خاصية لأى من الكينونتين التى تقوم بربطهما ببعض. وهذا التصور يأتى مطابقاً لما نجده من خصائص للعلاقات فى حياتنا اليومية، فلو نظرنا فى العلاقة الزوجية بين رجل وامرأة، على سبيل المثال، فإن تاريخ الزواج يعتبر خاصية للعلاقة الزوجية بين الطرفين وليس خاصية لأى منهما. كذلك هو الحال بالنسبة لتاريخ الملكية لعقار ما حيث إن تاريخ تملك العقار يعتبر خاصية لعلاقة التملك وليس خاصية لأى من المالك أو العقار. وفى مثل هذه الحالات تربط الخاصية (أو مجموعة الخصائص) بفئة العلاقة كما هو موضح فى الشكل رقم (٢-١١) لفئة العلاقة «مؤهل ل».

شكل رقم (٢-١١): تمثيل خصائص العلاقة فى مخطط كينونة - علاقة

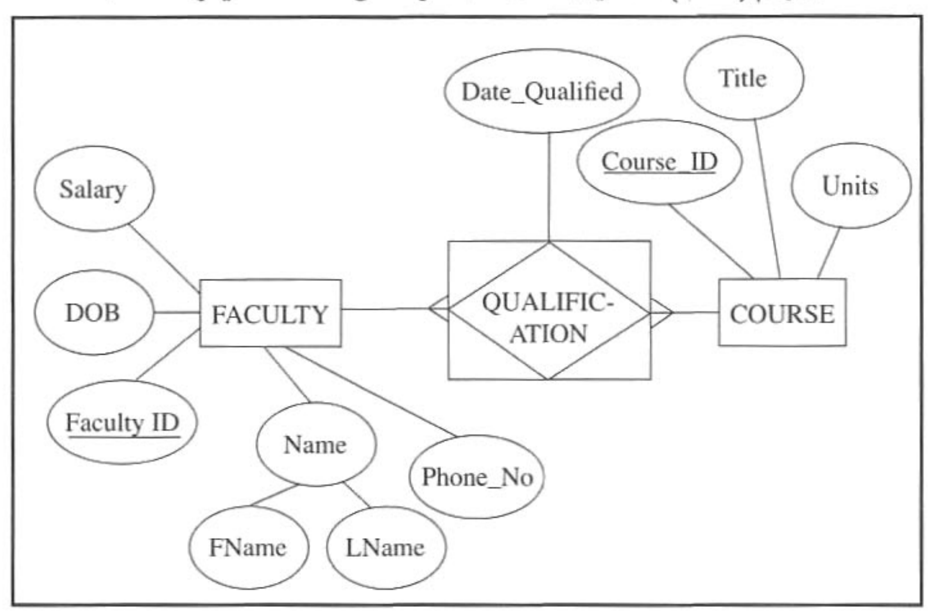


٢-١-٢-٢-٢ الكينونة المشاركة (Associative Entity):

إن ارتباط خاصية أو أكثر بفئة علاقة، كما فى حالة فئة العلاقة (مؤهل ل) فى الشكل (٢-١١)، قد تعنى أنه من الأنسب أن تمثل فئة العلاقة فى مخطط كينونة - علاقة على أنها من فئة كينونة. والكينونة المشاركة (Associative Entity) هى فئة علاقة تم تحويلها إلى فئة كينونة، وبذلك فهى تربط ما بين حالات فئتين من الكينونات (أو أكثر) وترتبط بالخصائص المتعلقة بها. ويمثل شكل العلاقة المشاركة بمستطيل

بداخله معين للدلالة على أن الكينونة قد كانت فى الأســـاس علاقة، ولكن تم تحويلها لعلاقة مشاركة. ويكتب بداخل الشكل اسم الكينونة المشاركة بحيث يكون اسماً مشتقاً من اســـم فئة العلاقة التى تم تحويلها، كما تبين العلاقة المشاركة فى الشكل رقم (٢-١٢) التى تم فيها تحويل فئة العلاقة (مؤهل ل) (Is_qualified) لتصبح علاقة مشـــاركة بمسمى (التأهيل) (Qualification).

شكل رقم (٢-١٢): تمثيل العلاقة المشاركة فى مخطط كينونة - علاقة



ويلاحظ فى المثال الســـابق عـــدم وجود فئة علاقة التى تمثل بالشـــكل المعين فى مخطـــط كينونة - علاقة بين الكينونة المشـــاركة والكينونتين الأخريين، وذلك لأن فئة الكينونة المشـــاركة تمثل العلاقة بين الكينونتين الأخريين وأنها فى الأســـاس كانت من فئـــة علاقة. كما يلاحظ أن كلتـــا التعدديتين (وهن من نوع متعدد) قد تمت إزاحتهما بحيث ينتهيان فى الكينونة المشاركة (أو كينونة الربط) عوضاً عن انتهائهما بالكينونتين الأخريين. ومن الأمور التى قد تستدعى تحويل فئة علاقة إلى فئة علاقة مشاركة (أو كينونة ربط) هو توافر بعض الشروط التالية (Hoffer et al, 2002):

١- أن تكون مشاركة كل فئة كينونة مرتبطة بفئة العلاقة من نوع متعدد.

٢- أن يكون لفئة العلاقة بعد تحويلها إلى كينونة مشـــاركة معنى مستقل ومعروف فى بيئة عمل المستفيدين، ومن المستحسن أن يكون لفئة الكينونة المشاركة مميز يتكون من خاصية واحدة فقط.

٣- أن يكون لفئة العلاقة خاصية أو أكثر بالإضافة إلى الخاصية التى ســتمثل مميزاً للكينونة المشاركة بعد عملية التحويل.

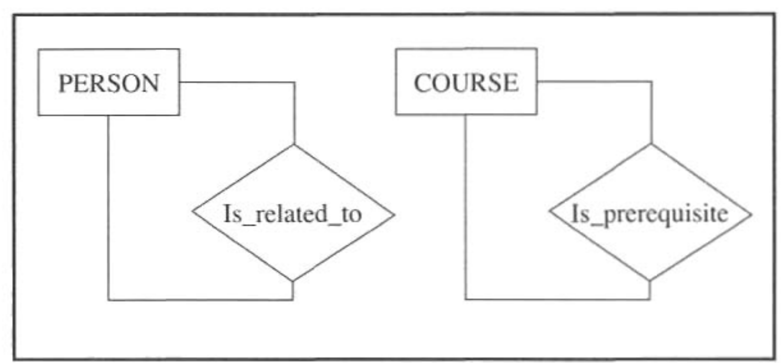
٤- أن ترتبط فئة الكينونة المشاركة (بعد عملية تحويل فئة العلاقة إلى كينونة مشاركة) بعلاقات مع كينونات أخرى غير تلك التى أدت إلى تكوين فئة الكينونة المشاركة.

٢-١-٢-٢-٣ درجة العلاقة (Degree of a Relationship):

درجـــة العلاقة هى عدد فئات الكينونات المرتبطـــة فى فئة العلاقة، ومنها العلاقة الأحادية، والعلاقة الثنائية والعلاقة الثلاثية. وعلى الرغم من أنه يمكن تمثيل علاقات ذات درجات أعلى، إلا أنه قلما توجد مثل هذه العلاقات على أرض الواقع. وفيما يلى أمثلة لكل نوع من درجات العلاقات:

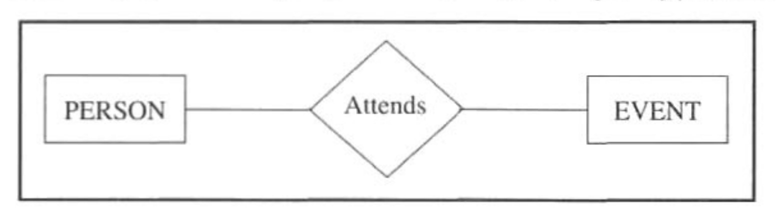
١- **علاقة أحادية (Unary Relationship)**: العلاقة الأحادية هى علاقة تربط بين حالات الكينونة نفســها. وتمثل العلاقة الأولى الموضحة فى الشــكل رقم (٢-١٣) «صلة القرابة» (Is_related_to) بين الأشخاص. ولكون العلاقة تربط بين حالات نفس فئة الكينونـــة (PERSON) فإن العلاقة أحادية (تشـــترك فيها فئة كينونة واحدة فقط). فعلى ســبيل المثال قد يكون شـــخص ما وليكـــن (ش١) مرتبطاً بعلاقـــة قرابة مع مجموعة من الأشخاص مثل (ش٢، ش٣، ش٤) وكل شخص من (ش٢، ش٣، ش٤) قد يكون مرتبطاً بعلاقات قرابة مع (ش١) وأشـــخاص آخرين كذلك. كذلك هو الحال بالنســبة للعلاقة الأحادية الثانية الموضحة فى نفس الشــكل التى تمثل المتطلبات الدراســية للمواد الدراســية المختلفة فى الجامعة الأهلية، إذ إن المادة الدراســية الواحدة قد تكون متطلباً لمواد دراســية أخرى، والمادة الدراســية نفسها قد تتطلب انتهاء الطالب من مجموعة مواد قبل أن يتمكن من التسجيل فى المادة. ومن ثم فإن هذه العلاقة تربط بين حالات نفس فئة كينونة المواد الدراسية (COURSE).

شكل رقم (٢-١٣): تمثيل العلاقة الأحادية فى مخطط كينونة - علاقة



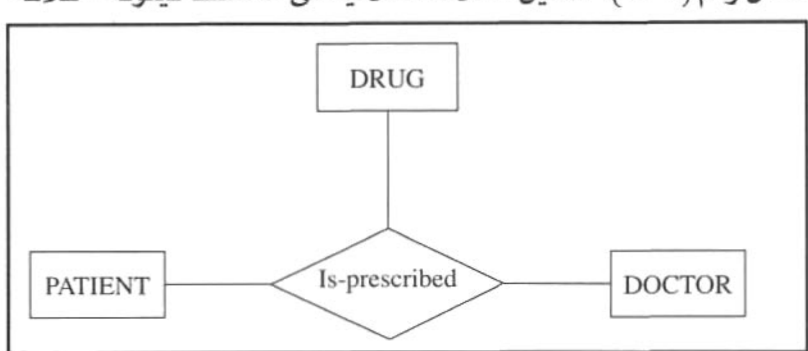
٢- **علاقة ثنائية (Binary Relationship):** العلاقة الثنائية تربط بين حالات فئتين من الكينونات. وتمثل العلاقة الثنائية فى الشكل رقم (٢-١٤) علاقة حضور الشخص لمناسبة معينة. ولكون علاقة الحضور (Attends) تربط بين الحالات التابعة لنوعين مختلفين من فئات الكينونات، وهما كينونة الشخص (PERSON) وكينونة المناسبة (EVENT)، فإن درجة العلاقة ثنائية.

شكل رقم (٢-١٤): تمثيل العلاقة الثنائية فى مخطط كينونة - علاقة



٣- **علاقة ثلاثية (Ternary Relationship):** العلاقة الثلاثية تربط بين حالات ثلاث فئات من الكينونات فى وقت واحد. وتمثل العلاقة فى الشكل رقم (٢-١٥) علاقة الوصفة الطبية للمريض حيث يشارك فى العلاقة ثلاث فئات من الكينونات وهى: المريض (PATIENT)، والطبيب (DOCTOR)، والدواء (DRUG). ومعنى «بنفس الوقت» أنه لا يمكن أن توجد حالة من حالات فئة علاقة الوصفة الطبية (Is_prescribed) دون مشاركة حالة من حالات الطبيب (وهو الذى وصف العلاج) مع حالة من حالات المريض (الذى سيتعاطاه) مع حالة من حالات الدواء (الذى تم وصفه).

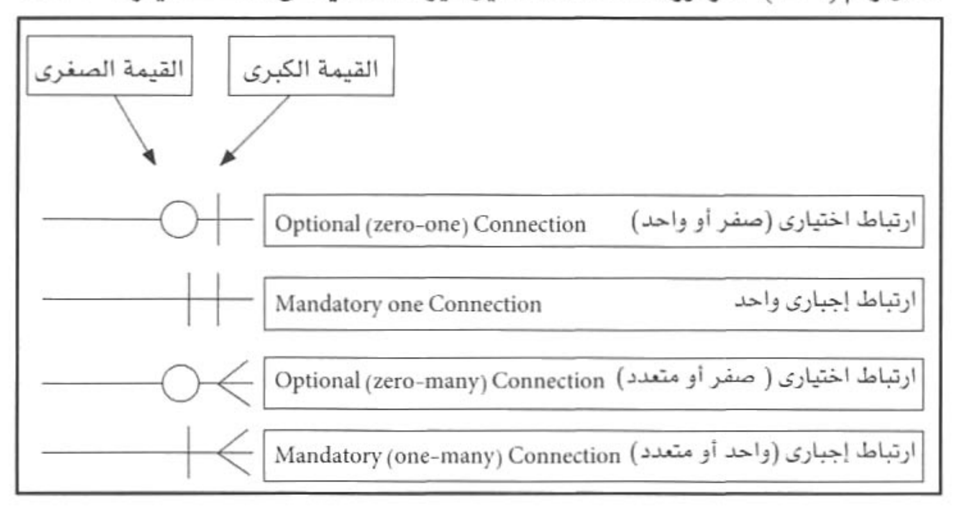
شكل رقم (٢-١٥): تمثيل العلاقة الثلاثية فى مخطط كينونة - علاقة



٢-١-٢-٢-٤ قيود التعددية (Cardinality Constraints):

تحدد قيود التعددية فى نموذج كينونة - علاقة عدد الحالات التى يجب أو يمكن ارتباطها فى كينونة ما مع كل حالة فى كينونة أخرى. وتمثل قيود التعددية من خلال القيمة الصغرى والقيمة الكبرى للعلاقة حيث تمثل القيمة الصغرى للعلاقة أقل عدد من الحالات التى يجب أو يمكن ارتباطها فى كينونة ما مع كل حالة فى كينونة أخرى، فى حين تمثل القيمة الكبرى أكبر تعددية للعلاقة أو بمعنى آخر أكبر عدد من الحالات التى يجب أو يمكن ارتباطها فى كينونة ما مع كل حالة فى الكينونة الأخرى. وعند تمثيل قيود التعددية تُستخدم الرموز الموضحة فى الشكل رقم (٢-١٦).

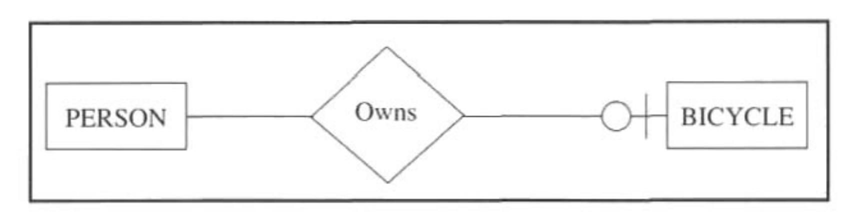
شكل رقم (٢-١٦): الرموز المستخدمة لتمثيل قيود التعددية فى مخطط كينونة - علاقة



وفيما يلى مثال لكل من الأنواع الأربعة من قيود التعددية فى نموذج كينونة - علاقة.

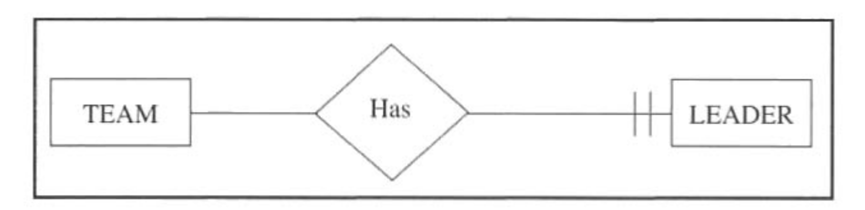
١- اختيـارى واحـد (Optional One): تعنـــى العلاقة فى الشـــكل رقم (٢-١٧) أن الحالـــة الواحدة فى كينونة شـــخص (PERSON) قد لا ترتبط بأية حالة فى كينونة دراجـــة (BICYCLE) أو أنها قد ترتبط بحالة واحدة على الأكثر. وبذلك فإن معنى العلاقة يصبح «قد لا يمتلك الشـــخص الواحد أية دراجة، ولكنه فى حالة امتلاكه لدراجـــة فإنه يمتلك دراجة واحـــدة على الأكثر». ولكونه ليـــس من الضرورى أن يمتلك كل شخص لدراجة فإن هذه العلاقة تعتبر اختيارية.

شكل رقم (٢-١٧): تمثيل تعددية اختيارى واحد فى مخطط كينونة - علاقة



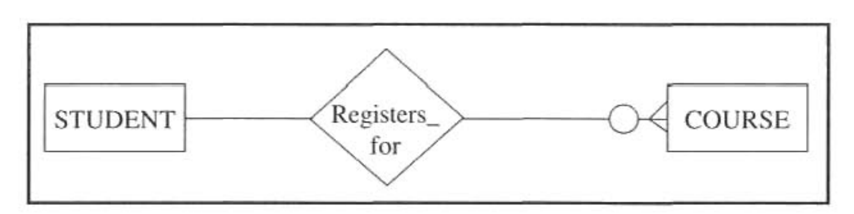
٢- إجبـارى واحـد (Mandatory One): تعنـــى العلاقة فى الشـــكل رقم (٢-١٨) أن الحالـــة الواحدة فى كينونة فريق (TEAM) يجب أن ترتبط بحالة واحدة فى كينونة قائـــد (LEADER) وحالة واحدة فقط (كحدٍ أعلى). وبذلك فإن معنى العلاقة بين كينونة الفريق وكينونـــة القائد يصبح «لكل فريق قائد، وقائد الفريق واحد فقط». ولكونه لا بد أن يكون لكل فريق قائد واحد (حدّاً أدنى)، فإن العلاقة ما بين كينونة فريق وكينونة قائد علاقة إجبارية.

شكل رقم (٢-١٨): تمثيل تعددية إجبارى واحد فى مخطط كينونة - علاقة



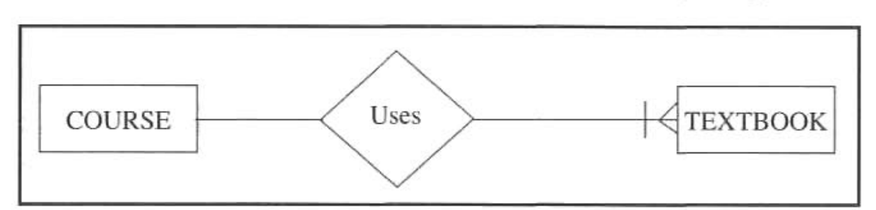
٣- اختيارى متعدد (Optional Many): تعنى العلاقة فى الشكل رقم (٢-١٩) أن الحالة الواحدة فى كينونة طالب (STUDENT) قد لا ترتبط بأية حالة فى كينونة مادة دراسية (COURSE) أو أنها قد ترتبط بأكثر من حالة واحدة. وبذلك فإن معنى العلاقة يصبح «قد لا يسجل الطالب الواحد فى أية مادة دراسية أو أنه قد يسجل فى أكثر من مادة دراسية». ولكونه ليس من الضرورى أن يسجل الطالب فى أية مادة فإن العلاقة اختيارية. كما أن العلاقة متعددة لكون الطالب قد يسجل فى أكثر من مادة دراسية. وبذلك تصبح العلاقة اختيارية متعددة.

شكل رقم (٢-١٩): تمثيل تعددية اختيارى متعدد فى مخطط كينونة - علاقة



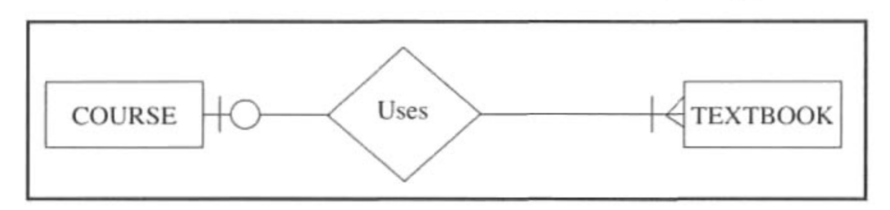
٤- إجبارى متعدد (Mandatory Many): تعنى العلاقة فى الشكل رقم (٢-٢٠) أن الحالة الواحدة فى كينونة مادة دراسية (COURSE) يجب أن ترتبط بحالة واحدة فى كينونة كتاب دراسى (TEXTBOOK) على الأقل، وقد ترتبط بأكثر من حالة فى كينونة كتاب دراسى. وبذلك فإن معنى العلاقة ما بين كينونة المادة الدراسية والكتاب الدراسى يصبح «لكل مادة دراسية كتاب واحد على الأقل، وقد يكون للمادة الدراسية أكثر من كتاب دراسى». ولكونه لا بد أن يكون لكل مادة دراسية كتاب واحد (حدّاً أدنى)، فإن العلاقة بين كينونة مادة دراسية وكينونة كتاب دراسى علاقة إجبارية. كما أن العلاقة متعددة لكونه قد يكون للمادة الدراسية الواحدة أكثر من كتاب دراسى واحد.

شكل رقم (٢-٢٠): تمثيل تعددية إجبارى متعدد فى مخطط كينونة - علاقة



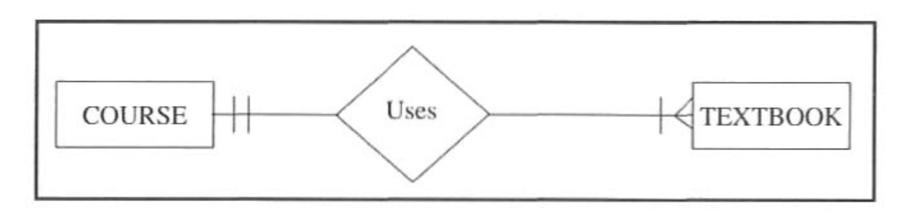
ولأن العلاقــة بين الكينونــات المرتبطة بها علاقة متبادلة؛ فإنــه يجب إدراج قيود تعدديــة العلاقة فــى كل اتجاهاتها، بمعنى أننــا قد نظرنا فى الأمثلة الســابقة إلى العلاقات فى اتجاه واحد فقط وهو من الجهة اليسرى من العلاقة إلى الجهة اليمنى. ولإدراج العلاقــة فى الاتجاه الآخر للمثال الأخير، على ســبيل المثال، إذا ما علمنا أن قاعدة العمل تنص على أنه قد لا يخصص الكتاب الدراســى لمادة دراســية، ولكنه قد يستخدم من قبل مادة دراسية واحدة (على الأكثر)، فإن العلاقة تصبح اختيارية واحدة كما هو موضح فى الشكل رقم (٢-٢١).

شكل رقم (٢-٢١): تمثيل تعددية العلاقة اختيارى واحد وإجبارى متعدد



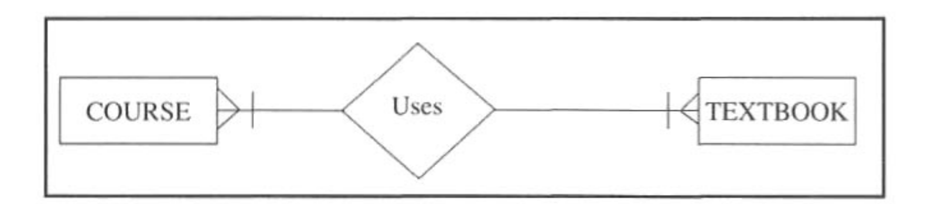
أما إن نصت قاعدة العمل على أن كل كتاب دراســى مخصص لمادة دراسية واحدة فقط، فإن العلاقة تصبح إجبارية واحدة كما فى الشكل (٢-٢٢).

شكل رقم (٢-٢٢): تمثيل تعددية العلاقة إجبارى واحد وإجبارى متعدد



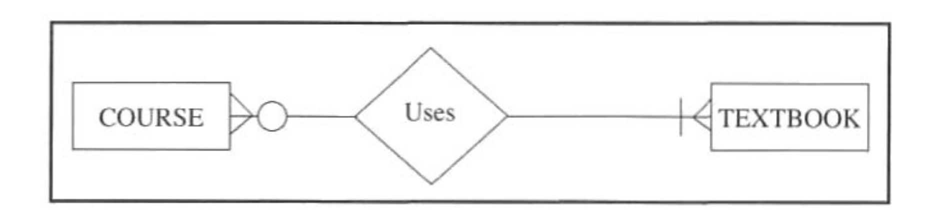
وفــى حال نصت قاعدة العمل على أن كل كتاب دراســى مخصص لمادة دراســية واحــدة على الأقل، وقد يخصص الكتاب لأكثر من مادة دراســية، فإن العلاقة تصبح إجبارية متعددة كما فى الشكل رقم (٢-٢٣).

شكل رقم (٢-٢٣): تمثيل تعددية العلاقة إجبارى متعدد وإجبارى متعدد



والاحتمـــال الأخير المتبقى للعلاقة هو فى حال نصت قاعدة العمل على أن الكتاب الواحد قد لا يخصص لأية مادة دراســية، وقد يخصص لأكثر من مادة دراســية. فى هذه الحالة تصبح العلاقة اختيارية متعددة كما فى الشكل رقم (٢-٢٤).

شكل رقم (٢-٢٤): تمثيل تعددية العلاقة اختيارى متعدد وإجبارى متعدد



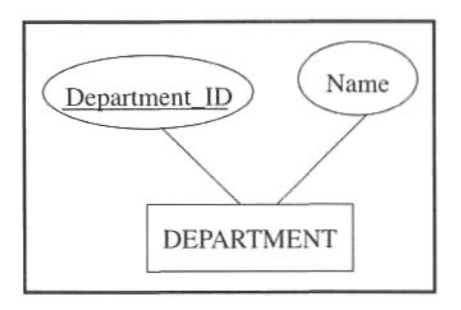
٢-١-٢-٣ حالة تطبيقية:

بعد اســتعراضنا للمفاهيم الأساسية المكونة لنموذج كينونة - علاقة، نعود لقواعد العمل المعمول بها فى الجامعة بهدف رسم مخطط كينونة - علاقة الذى يمثلها.

١- يوجد فى الجامعة عدد من الأقسام الدراسية، ولكل قسم (Department) من الأقسام العلمية رمز (Department_ID) يميزه عن بقية الأقسام، واسم (Name).

لقد سبق أن مثلنا قاعدة العمل هذه بفئة كينونة القسم الدراسى والخواص التابعة لها مع تحديد الخاصية المميزة وكان تمثيلنا للقسم الدراسى كما فى الشكل رقم (٢-٢٥).

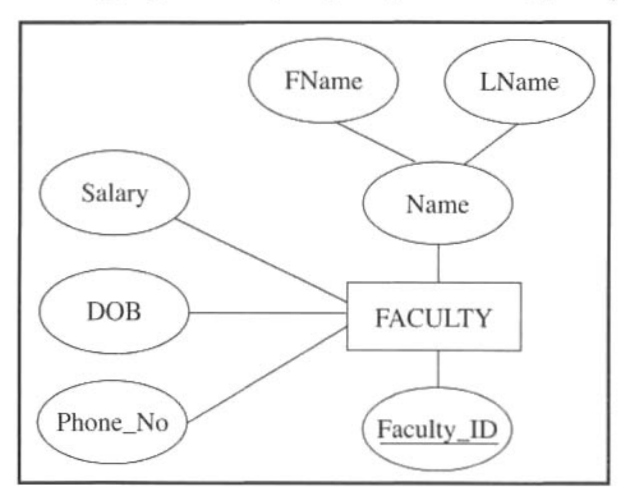
شكل رقم (٢-٢٥): تمثيل قاعدة العمل الأولى للجامعة الأهلية فى مخطط كينونة - علاقة



٢- يعمل فى الجامعة عدد من أعضاء هيئة التدريس، ولكل عضو هيئة تدريس (Faculty) رقـــم (Faculty_ID) يميزه عن بقية أعضاء هيئة التدريس، واســـم (Name) يتكون من (الاسم الأول (FName) واسم العائلة (LName)). وراتب شهرى (Salary)، وتاريخ ميلاد (Date of Birth (DOB))، ورقم هاتف (Phone_No).

لقد سبق أن مثلنا قاعدة العمل هذه بفئة كينونة عضو هيئة التدريس والخواص التابعة لها مع تحديد الخاصية المميزة وكان تمثيلنا لفئة الكينونة كما فى الشكل رقم (٢-٢٦).

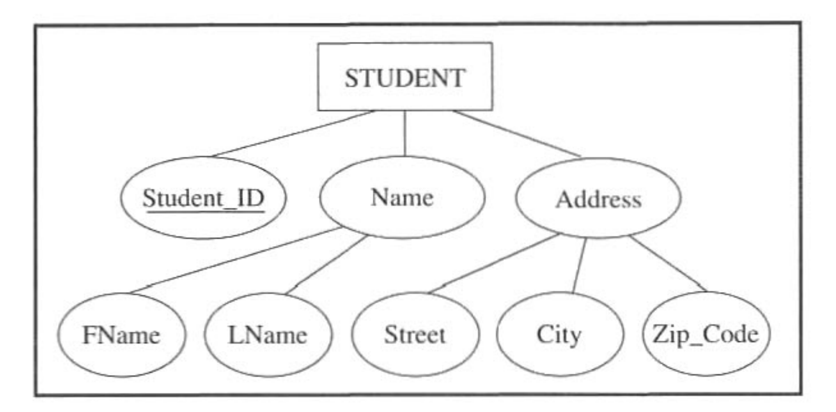
شكل رقم (٢-٢٦): تمثيل قاعدة العمل الثانية للجامعة الأهلية فى مخطط كينونة - علاقة



٣- يدرس فى الجامعة عدد من الطلاب، ولكل طالب (Student) رقم (Student_ID) يميزه عـــن بقية الطلاب فى الجامعة، واســـم (Name) يتكون من (الاســـم الأول (FName)

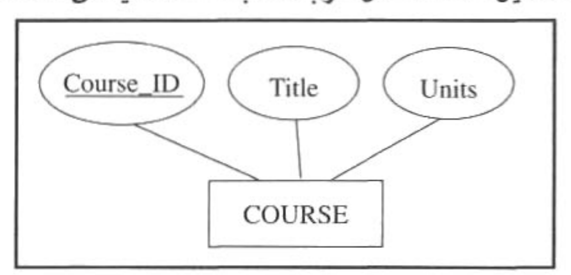
واسم العائلة (LName))، وعنوان بريدى (Address) يتكون من (اسم الشارع (Street)، واسم المدينة (City)، والرمز البريدى (Zip_Code)).

شكل رقم (٢-٢٧): تمثيل قاعدة العمل الثالثة للجامعة الأهلية فى مخطط كينونة - علاقة



٤- تنفذ الجامعة مجموعة من المواد الدراسية، ولكل مادة دراسية (Course) رمز (Course_ID) يميزها عن بقية المواد الدراسية التى تنفذها الجامعة، واسم (Title)، وعدد وحدات (أو ساعات) دراسية (Units).

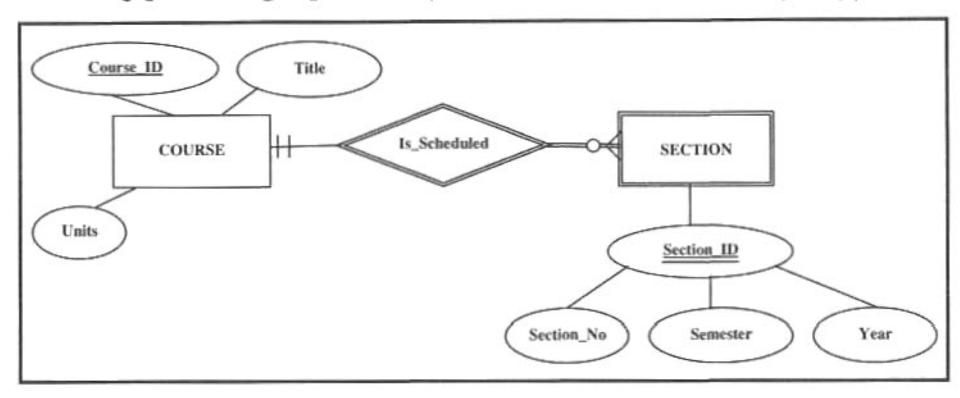
شكل رقم (٢-٢٨): تمثيل قاعدة العمل الرابعة للجامعة الأهلية فى مخطط كينونة - علاقة



٥- تنفذ (أو تعقد) كل مادة دراسية من خلال مجموعة (أو شعبة) دراسية (Section) واحدة أو أكثر فى الفصل الدراسى الواحد، أو قد لا تنفذ (أو تعقد) أية مجموعة (أو شعبة) للمادة الدراسية فى فصل دراسى معين، ولكل مجموعة دراسية رمز (Section_ID) يتكون من (رقم المجموعة، والفصل الدراسى المنفذة فيه (Semester)، والسنة الدراسية المنفذة فيها (Year)). أما رقم المجموعة (Section_No) فهو رقم (مثل ١، ٢، ٣، ... إلخ) يميز المجموعة عن بقية المجموعات المنفذة للمادة الدراسية

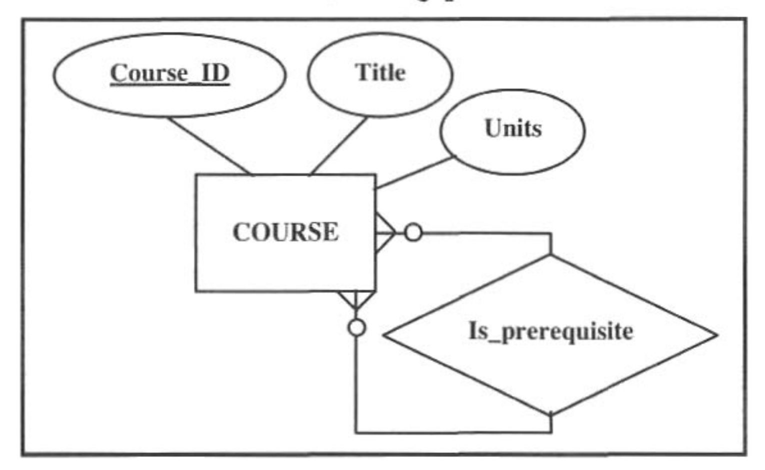
نفسها (وفى نفس الفصل والسـنة الدراسيين)، ولكنه لا يميزها بشكل منفرد عن بقية المجموعات الدراسية المنفذة للمواد الدراسية الأخرى فى الجامعة.

شكل رقم (٢-٢٩): تمثيل قاعدة العمل الخامسة للجامعة الأهلية فى مخطط كينونة - علاقة



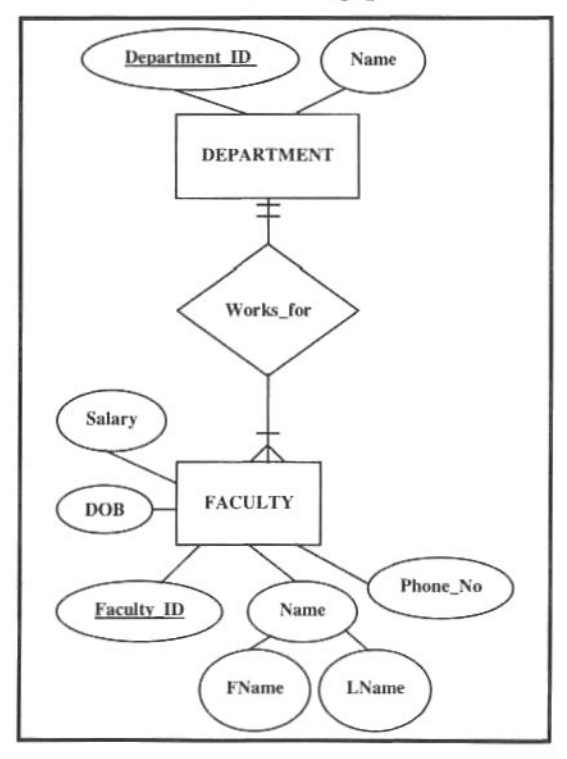
٦- قد يكون للمادة الدراسية الواحدة مجموعة من المتطلبات الدراسية، أو قد لا يكون للمادة الدراسـية أية متطلبات دراسـية. كما أن المادة الدراسية الواحدة قد تكون متطلباً لأكثر من مادة دراسية أو قد لا تكون متطلباً لأية مادة دراسية.

شكل رقم (٢-٣٠): تمثيل قاعدة العمل السادسة للجامعة الأهلية فى مخطط كينونة - علاقة



٧- يعمل (works for) فى كل قسم من أقسام الجامعة عضو هيئة تدريس واحد أو أكثر، وكل عضو من أعضاء هيئة التدريس يعمل فى قسم دراسى واحد فقط.

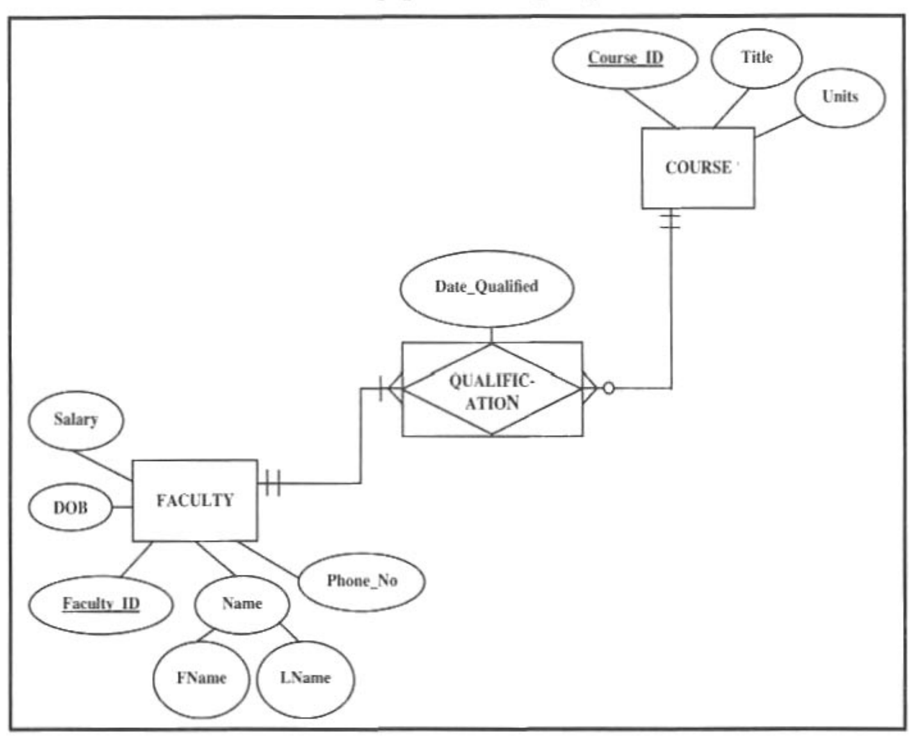
شكل رقم (٢-٣١): تمثيل قاعدة العمل السابعة للجامعة الأهلية فى مخطط كينونة - علاقة



٨- كل عضو هيئة تدريس فى الجامعة مؤهل (Qualified) لتدريس مادة دراسية واحدة على الأقل، وقد يتوافر للمادة الدراسية الواحدة أكثر من عضو هيئة تدريس مؤهلٍ لتدريسها، وقد لا يوجد من أعضاء هيئة التدريس من هو مؤهل لتدريس المادة.

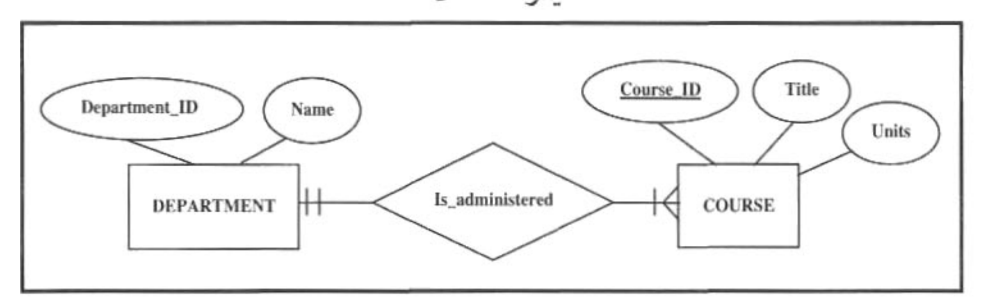
٩- عندما يتأهل عضـــو هيئة التدريس لتدريس مادة ما لأول مرة، يكون هنالك تاريخ لتأهيلــه (Qualification date) يحدد تاريخ تأهل عضو هيئة التدريس لتدريس المادة الدراسية.

شكل رقم (٢-٣٢): تمثيل قاعدة العمل الثامنة وقاعدة العمل التاسعة للجامعة الأهلية فى مخطط كينونة - علاقة



١٠- تدار (Administered) كل مادة دراسية من قبل قسم دراسى واحد من أقسام الجامعة، ويُدِير كل قسم مادة دراسية واحدة على الأقل.

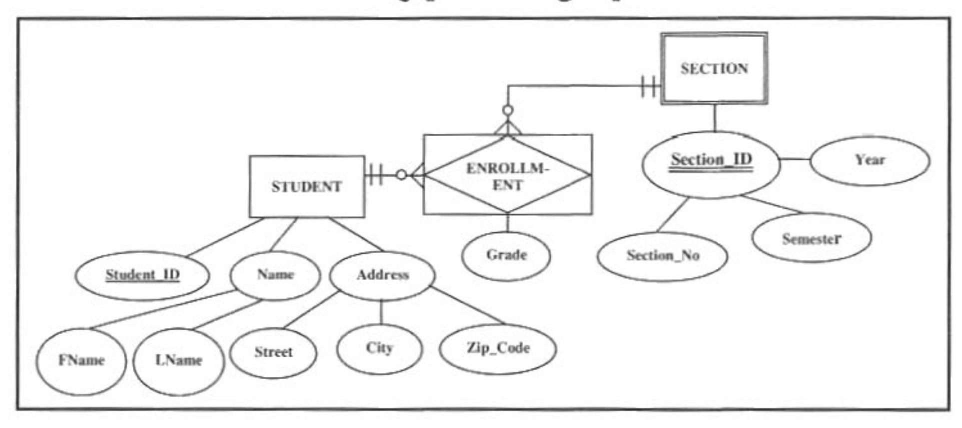
شكل رقم (٢-٣٣): تمثيل قاعدة العمل العاشرة للجامعة الأهلية فى مخطط كينونة - علاقة



١١- قد يســـجل (Enrolls) الطالب الواحد فى أكثر من مجموعة (أو شعبة) دراسية أو قد لا يسجل فى أية مجموعة (أو شعبة) دراسية، والمجموعة (أو الشعبة) الدراسية الواحدة قد لا يسجل فيها أى طالب أو قد يسجل فيها أكثر من طالب.

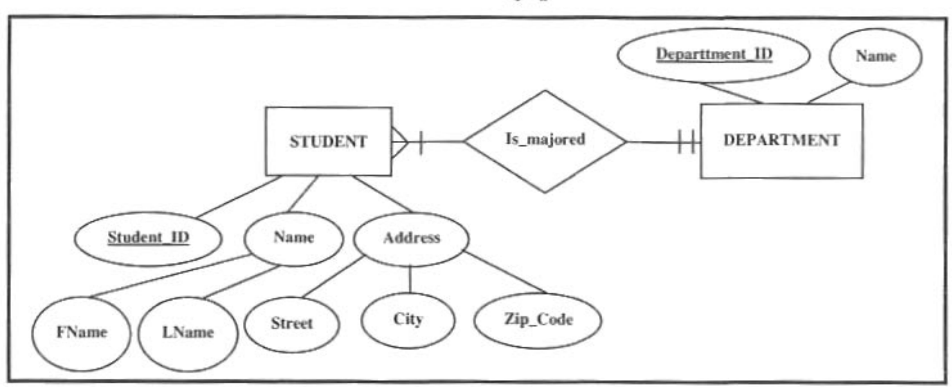
١٢- عندما يســـجل طالب فى مجموعة دراســـية تكون له درجـــة (Grade) تعطى عند انتهائه من الدراسة فى المجموعة.

شكل رقم (٢-٣٤): تمثيل قاعدة العمل الحادية عشرة والثانية عشرة للجامعة الأهلية فى مخطط كينونة - علاقة



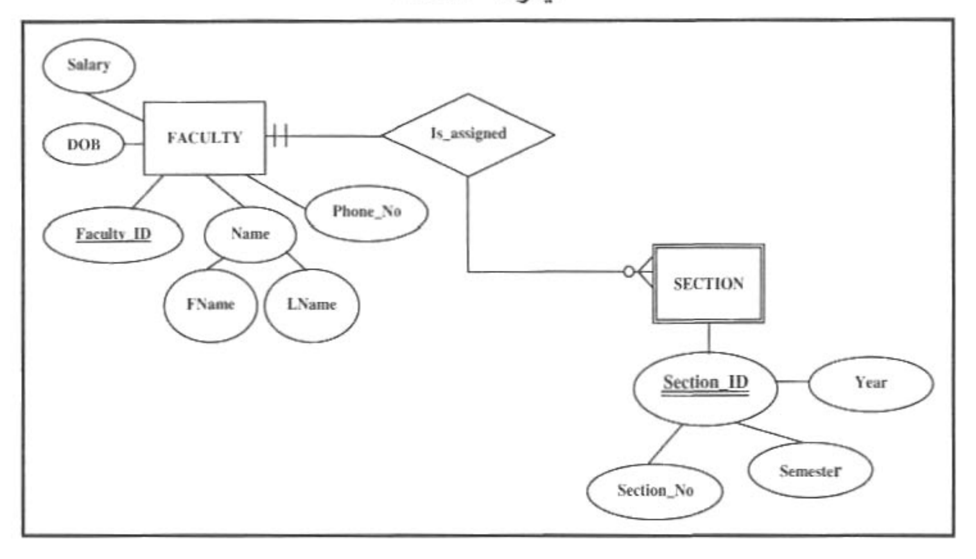
١٣- يتخصص كل طالب (Majors) فى قسم دراسى واحد فقط، ويتخصص فى القسم الدراسى الواحد أكثر من طالب.

شكل رقم (٢-٣٥): تمثيل قاعدة العمل الثالثة عشرة للجامعة الأهلية فى مخطط كينونة - علاقة



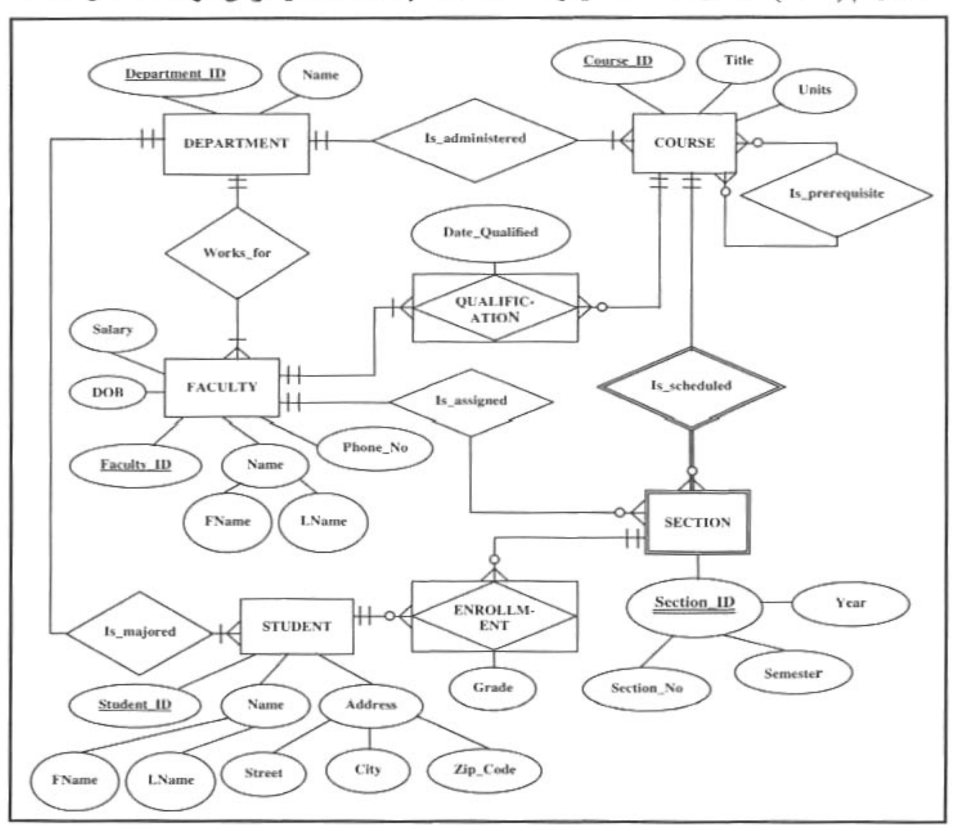
١٤- يكلف (Assigned) كل عضو هيئة تدريس بتدريس مجموعة (أو شعبة) دراسية واحدة أو أكثر، وقد لا يكلف عضو هيئة التدريس بأية مجموعة (أو شعبة) دراسية، والمجموعة (أو الشعبة) الدراسية الواحدة تكلف لعضو هيئة تدريس واحد فقط.

شكل رقم (٢-٣٦): تمثيل قاعدة العمل الرابعة عشرة للجامعة الأهلية فى مخطط كينونة - علاقة



ويمثل الشكل رقم (٢-٣٧) كامل مخطط كينونة-علاقة وفق قواعد العمل المعمول بها فى الجامعة الأهلية المذكورة أعلاه.

شكل رقم (٢-٣٧): كامل مخطط كينونة - علاقة للجامعة الأهلية وفق قواعد العمل المعطاة



الفصل الثالث

نموذج كينونة - علاقة المطور

إن نمـــوذج كينونـــة - علاقة الذى تم اســـتعراض مكوناته فى الفصل الســـابق قد لاقـــى قبولاً كبيراً من مصممـــى نظم قواعد البيانات منذ بدايـــة ظهوره فى منتصف الســـبعينيات الميلادية؛ وذلك لقدرته على نمذجة غالبية قواعد العمل المعمول بها فى المنظمات الحديثة، إلا أن النموذج بشـــكله المبدئى لا يســـتطيع أن يقوم بنمذجة كافة قواعـــد العمل المعمول بها فى غالبية المنظمـــات حالياً. وقد حدا هذا بالباحثين إلى تطوير المكونات الرئيســـية للنموذج حتى يتمكن من نمذجة أكبر قدر ممكن من قواعد العمل المعمول بها فى المنظمات المختلفة. ويســـتعرض هذا الفصل من الكتاب نموذج «كينونـــة - علاقة المطـــور» (Enhanced Entity-Relationship Model). ومن أهم المكونات الرئيسية فى النموذج المطور هى علاقة الأنواع الرئيسية بالأنواع الفرعية (/Supertype Subtype Relationship) ومبدأ التجميع (Aggregation).

٣-١ الأنواع الرئيسية والأنواع الفرعية (Supertype/Subtype) فى نموذج كينونة- علاقة المطور:

إن مـــن أهم مكونـــات نموذج كينونة - علاقة المطور هو الأنواع الرئيســـية والأنواع الفرعية. ويمكننا هذا المكون الجديد من نمذجة نوع عام واحد من فئة كينونة، يسمى النوع الرئيســـى، بحيث يتم تقسيمه إلى عدة أنواع مخصصة، تسمى الأنواع الفرعية. ويتكون النوع الرئيســـى، كما هو الحال فى فئات الكينونات الأخرى، من مجموعة من الخصائص المشـــتركة فى كافة الكينونات الفرعية المخصصة من النوع الرئيسى. كما أن النوع الرئيســـى قد يرتبط بعلاقات مع فئـــات كينونات أخرى. ويتم توريث كلٍّ من خصائص النوع الرئيسى والعلاقات التى يرتبط بها لكل نوع من الأنواع الفرعية، هذا بالإضافـــة لما قد يكون للأنواع الفرعية من خصائص (أو علاقات) تميزها عن الأنواع الفرعية الأخرى التى تتبع لنفس النوع الرئيسى.

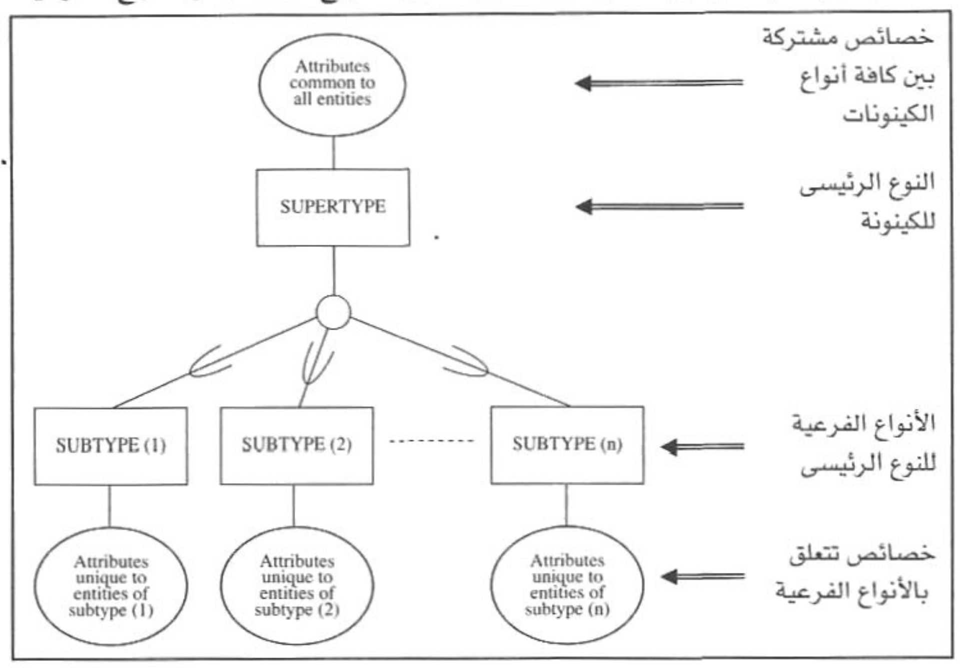
إن الأنـــواع الفرعيـــة ما هى إلا مجموعات جزئية لأحد الأنواع الرئيســـية. فعلى ســـبيل المثال، يمكن تمثيل فئة كينونة «طالـــب» (STUDENT) بحيث يتفرع منها نوعان

فرعيان هما فئة طلبة «الدراسات العليا» (Graduate_Students) وفئة «طلبة البكالوريوس» (Undergraduate_Students). ويلاحظ في كلا النوعين الفرعيين أن لهما خصائص مشتركة كما أن كلاً منهما له خصائص لا يشترك فيها مع النوع الفرعي الآخر. فعلى سبيل المثال، رقم الطالب، وعنوان الطالب، ورقم هاتف الطالب، تمثل بعض الخصائص المشتركة لكافة الطلبة بغض النظر عن كونهم من طلبة الدراسات العليا أو طلبة البكالوريوس. على النقيض من ذلك فإن فئة طلبة الدراسات العليا قد يكون لها خصائص تميزها عن طلبة دراسات البكالوريوس مثل رقم مكتب الطالب (إذ إن غالبية الجامعات توفر مكاتب لطلبة الدراسات العليا)، أو علاقات خاصة بهذه الفئة مثل ارتباطها بكينونة أعضاء هيئة التدريس لتمثيل علاقة المشرف الدراسي لمجال بحث الطالب. والسؤال الذى قد يطرح، لماذا لا نمثل كل فئة من الكينونات الفرعية السابقة بشكل أبسط من خلال نمذجتها على أساس أنها فئات لكينونات مختلفة؟ إن الإجابة عن هذا التساؤل ليست ببساطة السؤال ذاته؛ لأنها تعتمد على ما تريده المنظمة من النموذج، إذ إنه من التحديات التى تواجه عملية نمذجة البيانات التعرف على الكينونات المتشابهة فيما بينها بشكل كبير وتمثيلها بشكل واضح ضمن فئة واحدة. وإن استخدام الأنواع الرئيسية تمكننا من ذلك، وفى الوقت نفسه تمكننا من تخصيص بعض الأنواع الفرعية التى تتميز بخصائص لا تشترك فيها مع الأنواع الفرعية الأخرى، خاصة إذا كانت هذه الأنواع الفرعية ذات معنى فى بيئة المنظمة، مثل طلبة الدراسات العليا وطلبة البكالوريوس ويجب إيضاحها ضمن نموذج البيانات.

٣-١-١ المفاهيم الأساسية والرموز المستخدمة فى الأنواع الرئيسية والأنواع الفرعية:

يوضح الشكل رقم (٣-١) الرموز الأكثر شيوعاً فى تمثيل الأنواع الرئيسية والأنواع الفرعية (Hoffer et al, 2002)، حيث يتصل النوع الرئيسى بخط مستقيم مع دائرة يتفرع منها خط مستقيم لكل نوع فرعى من أنواع النوع الرئيسي. أما الرمز (U) الموضوع على الخط الواصل بين نقطة التفرع (وهى الدائرة) إلى كل نوع فرعى فيقصد منها أن النوع الفرعى هو مجموعة جزئية من النوع الرئيسي، بالإضافة لكونها تبين اتجاه العلاقة بين النوع الرئيسي والنوع الفرعي. وترتبط الخصائص المشتركة لكافة أنواع الكينونات، ويتضمن ذلك «المعرف» (Identifier)، بالنوع الرئيسي. أما الخصائص المتعلقة بنوع فرعى ما دون غيره من الأنواع الفرعية فترتبط بالنوع الفرعى نفسه.

شكل رقم (٣-١): الرموز الأكثر شيوعاً فى تمثيل الأنواع الرئيسية والأنواع الفرعية



ولتوضيح مفهوم الأنواع الفرعية والأنواع الرئيسية، لنفترض المثال البسيط التالى الذى يعد من أكثر الأمثلة شيوعاً، وهو وجود منظمة يعمل فيها ثلاثة أنواع من الموظفين: موظفون يعملون بأجر الساعة (Hourly Employees)، وموظفون مثبتون على وظائف رسمية بأجر شهرى (Salaried Employees)، وموظفون مستشارون يعملون وفق عقود تحدد أجرهم (Contract Employees). ومن الخصائص المهمة للأنواع الثلاثة من الموظفين ما يلى:

موظفو أجر الساعات (Hourly Employees): رقم الموظف (Employee_No)، واسم الموظف (Name)، وعنوان الموظف (Address)، وتاريخ التعيين (Date_Hired)، وأجر الساعة (Hourly_Rate).

موظفو الأجر الشهرى (Salaried Employees): رقم الموظف (Employee_No)، واسم الموظف (Name)، وعنوان الموظف (Address)، وتاريخ التعيين (Date_Hired)، والمرتب الشهرى (Monthly_Salary).

موظفو الأجر وفق عقود استشارية (Consultant Employees): رقم الموظف (Employee_No)، واسم الموظف (Name)، وعنوان الموظف (Address)، وتاريخ التعيين (Date_Hired)، ورقم العقد (Contract_No)، والتكلفة الاستشارية (Billing_Rate).

ويلاحظ أن الفئات الثلاث من الموظفين يشتركون فى خصائص مشتركة مثل رقم الموظف، وعنوان الموظف. كما أن كل فئة من الموظفين ترتبط بخاصية أو أكثر لا ترتبط فيها الفئتان الأخريان من فئات الموظفين، مثل: خاصية أجر الساعة بالنسبة لفئة الموظفين بأجر الساعة. وعندما نحاول نمذجة بيانات هذا المثال وفق نموذج كينونة - علاقة، يمكن النظر فى ثلاثة خيارات لنمذجته، وهى كالتالى:

١- تمثيل الفئات الثلاث من الموظفين من خلال فئة كينونة واحدة هى كينونة الموظفين (EMPLOYEE). وعلى الرغم من بساطة هذا النموذج إلا أن العيب الرئيسى لهذا الأسلوب هو أن كينونة الموظفين يجب أن تحتوى على كافة خصائص الفئات الثلاث من الموظفين. وعند عدم انطباق خاصية ما على إحدى كينونات الموظفين فإنها تترك دون استخدام أو تكون قيمتها غير معرفة (Null). وعندما تتم نمذجة هذا المثال وفق هذه الطريقة فإنها ستعقد عملية كتابة التطبيقات على قاعدة البيانات؛ لأن كل تطبيق يجب أن يكتب بطريقة يستطيع من خلالها التمييز بين الفئات الثلاث من الموظفين والتعامل معها بشكل سليم هذا بالإضافة لما قد ينتج عن هذه الطريقة من ضياع للمساحة التخزينية نتيجة للحقول غير المستخدمة من قبل كل كينونة من كينونات الفئة.

٢- تمثيل كل فئة من فئات الموظفين من خلال فئة كينونة خاصة بها بحيث ينتج عن هذه الطريقة ثلاث فئات من الكينونات. وعلى الرغم من سهولة هذه الطريقة أيضاً، إلا أن هذه الطريقة لا تمكن من التعرف على الخصائص المشتركة بين الفئات الثلاث من الموظفين. بالإضافة لذلك فعلى مستخدمى قاعدة البيانات توخى الدقة فى اختيار الفئات المناسبة عند تعاملهم مع قاعدة البيانات.

٣- تمثيل الخصائص المشتركة للفئات الثلاث من الموظفين من خلال نوع رئيسى واحد بمسمى «موظف» (EMPLOYEE) يتفرع منه ثلاثة أنواع فرعية هى: موظفو أجر الساعات (HOURLY_EMPLOYEE)، وموظفو الأجر الشهرى (SALARIED_EMPLOYEE)، وموظفو الأجر وفق عقود استشارية (CONSULTANT). وتمكن هذه الطريقة من التعرف على الخصائص المشتركة بين الفئات المختلفة من الموظفين، وفى الوقت نفسه، معرفة الخصائص التى تنفرد فيها كل فئة من فئات الموظفين.

ويمثل الشكل رقم (٣-٢) النوع الرئيسى للموظفين (EMPLOYEE) والأنواع الفرعية الثلاثة التى تمثل الفئات الثلاث من الموظفين بحيث تم ربط الخصائص المشتركة بين الفئات الثلاث من الموظفين (ومن ضمنها المعرف) بالنوع الرئيسى وربط الخصائص التى تتفرد فيها كل فئة من فئات الموظفين بالنوع الفرعى الذى يمثل فئة الموظفين.

شكل رقم (٣-٢): النوع الرئيسى لكينونة الموظفين وأنواعها الفرعية الثلاثة

Name
Address
Employee_No
EMPLOYEE
Date_Hired
HOURLY EMPLOYEE
SALARIED EMPLOYEE
CONSULTANT
Hourly_Rate
Monthly_Salary
Contract_No
Billing_Rate

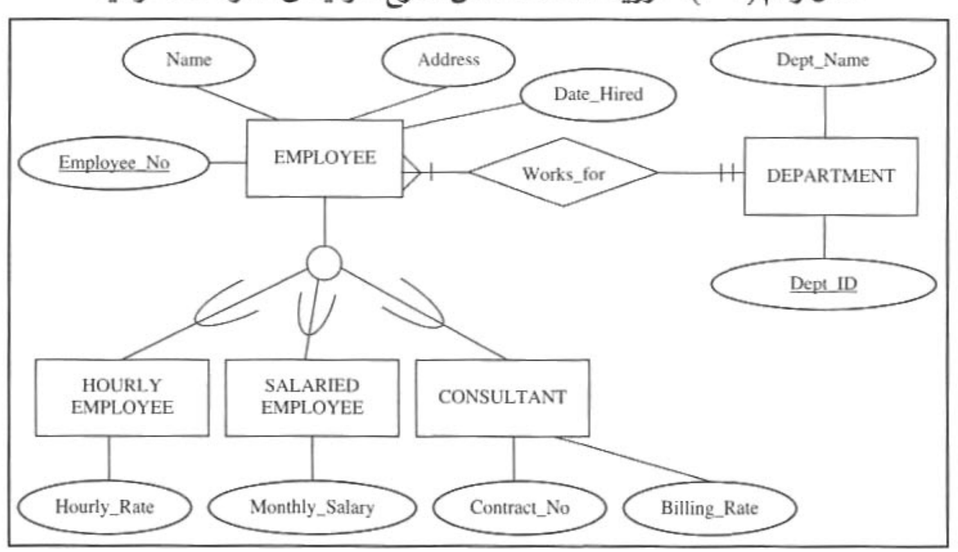
٣-١-١-١ توريث الخصائص والعلاقات:

إن أى نوع فرعى يعد فئة كينونة قائمة بذاتها. كما أن أية حالة من النوع الفرعى لا بد أن يقابلها نفس الحالة فى النوع الرئيسى. ويمكن تصور هذا الوضع على أساس أن كل حالة قد تم تقسيم خصائصها بين النوع الرئيسى (الذى ترتبط به كافة الخصائص المشتركة لكافة الكينونات بغض النظر عن الأنواع الفرعية التى تتبعها هذه الكينونات) والنوع الفرعى الذى تتبعه الحالة. فعلى سبيل المثال، لو افترضنا وجود حالة مستشار باسم «صالح الأحمد» (Saleh Al-Ahmad) ضمن النوع الفرعى (CONSULTANT)، فإنه لا بد من وجود نفس الشخص ضمن حالات النوع الرئيسى (EMPLOYEE). وبناء على ذلك فإن أية حالة من حالات أى نوع فرعى لابد أن تتصف، ليس بخصائص النوع الفرعى فحسب، وإنما بخصائص النوع الرئيسى الذى يتبعه النوع الفرعى كذلك.

ويعرف الوضع أعلاه باسم توريث الخصائص (Attribute Inheritance)، إذ إن كينونات النوع الفرعى ترث قيماً لكافة خصائص النوع الرئيسى. وبهذه الطريقة يصبح من غير الضرورى تكرار الخصائص التى ترتبط بالنوع الرئيسى فى الأنواع الفرعية التى تتبعه. فعلى سبيل المثال، إن اسم الموظف هو خاصية تتبع لفئة كينونة الموظفين ولا تتكرر ضمن أى من الأنواع الفرعية التى تتبعها. لذلك فإن اسم المستشار «صالح الأحمد» يعد خاصية من خصائص فئة كينونة الموظفين وليس من خصائص فئة الكينونة الفرعية (CONSULTANT)، إلا أن أجر المستشار «صالح الأحمد» خاصية تتبع فئة الكينونة الفرعية (CONSULTANT).

بالإضافة لتوريث الخصائص، فإن الأنواع الفرعية ترث العلاقات التى يرتبط بها النوع الرئيسى. فعلى سبيل المثال، لو افترضنا أن أية موظف لا بد أن يعمل فى إدارة واحدة فقط من إدارات المنظمة، كما هو موضح فى الشكل رقم (٣-٣)، فإن كافة الأنواع الفرعية لفئة كينونة الموظفين سترث هذه العلاقة بمعنى أن أى مستشار، أو موظف بأجر الساعة، أو موظف بالأجر الشهرى لابد أن يعمل فى إدارة واحدة فقط من إدارات المنظمة. ونظراً لكون هذه العلاقة مع فئة كينونة الإدارات علاقة تنطبق على كافة فئات الموظفين، فقد تم ربط هذه العلاقة بالنوع الرئيسى وليس بأى من الأنواع الفرعية.

شكل رقم (٣-٣): توريث العلاقات من النوع الرئيسى لأنواعه الفرعية



وبناء على ما سبق يمكن طرح السؤال التالى: متى يمكن استخدام علاقة النوع الرئيسى والأنواع الفرعية؟

يعتمد استخدام علاقة النوع الرئيسى والأنواع الفرعية على الحالة التى نحاول نمذجتها. وبشكل عام يقترح وجود أى من الحالتين التاليتين استخدام علاقة النوع الرئيسى والأنواع الفرعية:

١- وجود خصائص ترتبط ببعض الكينونات ولكن ليس بكافة أنواع الكينونات، مثل كينونة الموظفين التى تطرقنا إليها سابقاً.

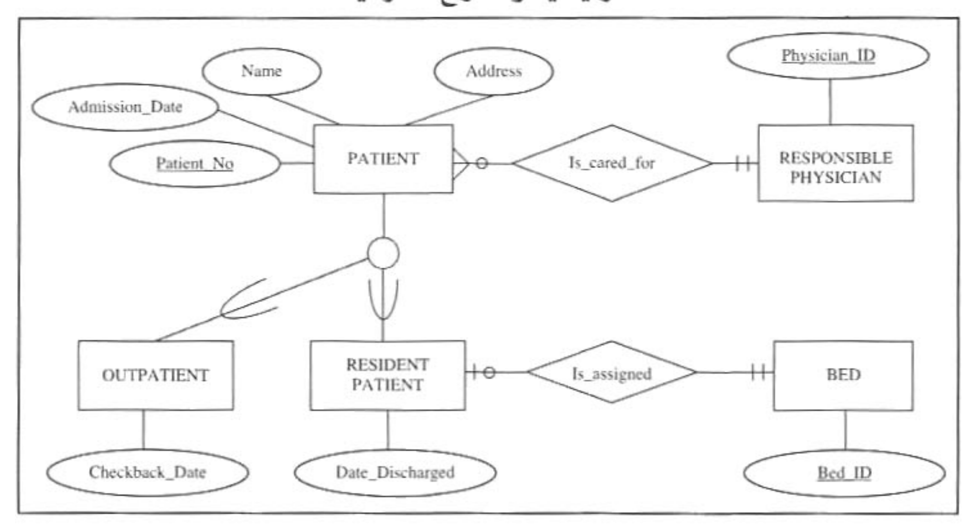
٢- ارتباط إحدى فئات الكينونات الفرعية بعلاقة لا ترتبط فيها أى من فئات الكينونات الفرعية الأخرى.

والشكل رقم (٣-٤) يحتوى على جزء من نموذج كينونة - علاقة لأحد المستشفيات يعد مثالاً لإيضاح كلتا الحالتين السابقتين، فكينونة «المريض» (PATIENT) هى نوع رئيسى يرتبط به نوعان فرعيان هما «المرضى المنومون» (RESIDENT_PATIENT) و «مرضى العيادات الخارجية» (OUTPATIENT). ويرتبط بالنوع الرئيسى كافة الخصائص المشتركة لكينونات المرضى وهى: اسم المريض، عنوان المريض، وتاريخ المراجعة، ورقم المريض (وهو معرف الكينونة). بالإضافة لذلك، فإن كل مريض يرتبط بعلاقة مع «طبيب مسئول» (RESPONSIBLE_PHYSICAIAN) عن علاج المريض. ولكون كافة المرضى مرتبطين بعلاقات مع أطباء مسئولين عن علاجهم بغض النظر عن كونهم مرضى منومين أو مرضى عيادات خارجية، فقد تم ربط هذه العلاقة مع الأطباء المسئولين من خلال النوع الرئيسى عوضاً عن أى من الأنواع الفرعية.

أما بالنسبة للأنواع الفرعية، فإن كلاً منها يرتبط بخاصية لا يرتبط بها النوع الآخر. فمرضى العيادات الخارجية يرتبطون بخاصية «تاريخ المراجعة التالية» (Checkback_ Date) التى لا يرتبط بها المرضى المنومون، والمرضى المنومون يرتبطون بخاصية «تاريخ الخروج» (Date_Discharged) التى لا يرتبط بها مرضى العيادات الخارجية. كما أن المرضى المنومين يرتبطون بعلاقة خاصة بهم وهى علاقة «مسند إلى» (Is_assigned). وهذه العلاقة تربط كل مريض منوم بالسرير الطبى الذى سيسند إليه أثناء فترة علاجه فى المستشفى. ونظراً لكون هذه العلاقة لا تنطبق على مرضى العيادات الخارجية، فإنه قد تم ربطها بالمرضى المنومين فقط. ويعنى هذا أن العلاقات الخاصة بالأنواع الفرعية لا تورث للأنواع الفرعية الأخرى ضمن نفس النوع الرئيسى، أو للنوع الرئيسى نفسه، وإنما تبقى خاصة بالنوع الفرعى الذى يرتبط بها فقط.

ومـــن خلال عملية توريث الخصائص، فإن أى مريض ســـواء كان منوماً أو مريضاً فى العيادات الخارجية سيرث كافة خصائص النوع الرئيسى دون الحاجة إلى تكرارها ضمـــن خصائصه. فعلى ســـبيل المثال، لكل مريض ســـواء كان منومـــاً أو من مرضى العيادات الخارجية رقماً، واسماً، وعنواناً، وتاريخاً للمراجعة يرثها من النوع الرئيسي. ومما ســـبق يتضح إن توريث الخصائص والعلاقات يتم من الأعلى إلى الأسفل، أى من النوع الرئيسى إلى أنواعه الفرعية، وأن عملية التوريث ليست انعكاسية بمعنى أن النوع الرئيســـى، أو الأنواع الفرعية الأخرى، لا ترث الخصائص والعلاقات التى يرتبط بها نوع فرعى ما.

شكل رقم (٣-٤): مثال لإيضاح الحالات التى يفضل فيها استخدام علاقات الأنواع الرئيسية والأنواع الفرعية



٣-١-١-٢ توصيف القيود فى علاقات الأنواع الرئيسية والأنواع الفرعية:

يوضح هذا الجزء الرموز المستخدمة لتوصيف القيود فى علاقات الأنواع الرئيسية والأنـــواع الفرعيـــة، إذ تمكننا هذه القيـــود من تمثيل بعض قواعـــد العمل المهمة عند اســـتخدام نمـــوذج كينونة - علاقة فـــى نمذجة بيانـــات المنظمة. وأهـــم نوعين من القيود هما قيد التخصيـــص (Specialization Constraint) وقيد الانفصال (Disjointness Constraint) (Elmasri and Navathe, 2004).

٣-١-١-٢-١ قيد التخصيص (Specialization Constraint):

لقد أشــرنا أعلاه إلى أن كل حالة موجودة فى أحد الأنواع الفرعية لابد أن يوجد ما يقابلها فى النوع الرئيســى. ولكن الســؤال هو: هل هذه العملية عكسية؟ بمعنى هل يجب أن تكون كل حالة موجودة فى النوع الرئيســى ممثلة ضمن إحدى حالات نوع فرعى واحد على الأقل؟

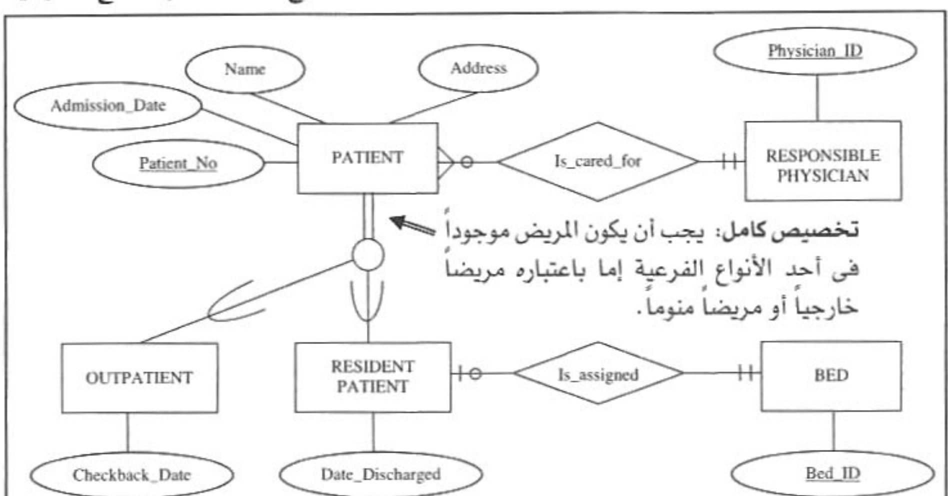
يجيب قيد التخصيص عن هذا التساؤل من خلال قاعدتين هما: قاعدة التخصيص الكامــل وقاعدة التخصيص الجزئــى. إن قاعدة التخصيص الكامل تعنى أن أى حالة موجودة فى النوع الرئيسى لا بد أن تمثل ضمن واحدة على الأقل من الأنواع الفرعية. أمــا القاعدة الثانيــة، قاعدة التخصيص الجزئى، فتعنى أنــه من الممكن لحالة ممثلة ضمن النوع الرئيسى أن لا تمثل ضمن أى من الأنواع الفرعية. وفيما يلى إيضاح لكلتا القاعدتين.

٣-١-١-٢-١-١ التخصيص الكامل (Total Specialization):

يمثل الشكل رقم (٣-٥) المثال السابق المتعلق بمرضى أحد المستشفيات بعد إيضاح رمــز التخصيص عليه. إن قاعدة العمل المعمــول بها فى هذا المثال هى كما يلى: إن أى مريض فى المستشــفى لا بــد أن يكون إما مريضاً منومــاً أو مريضاً فى العيادات الخارجية. ويعنى هذا أنه لا يوجد أى نوع من أنواع المرضى ســوى الفئتين السابقتين منهــم. كما يعنى هــذا أن قيد التخصيص هــو تخصيص كامل؛ فــأىُّ حالة لمريض موجود فى النوع الرئيســى لابد أن يوجد ما يقابلها فى أحد الأنواع الفرعية. ويمثل قيــد التخصيص الكامل بخطين مزدوجين يصلان بين النوع الرئيســى وهو «المريض» (PATIENT) ونقطة تفرع الأنواع الفرعية (وهى الدائرة).

ويعنــى هذا التمثيل للتخصيص أنه عندما تتم عملية إضافة مريض جديد لحالات كينونــة المرضى لا بد أن يضاف المريض إما للنوع الفرعى الذى يمثل المرضى المنومين أو النــوع الفرعى الذى يمثل مرضى العيادات الخارجيــة. وفى حالة إضافة المريض ضمن حالات المرضى المنومين فإنه لا بد أيضاً من إضافة حالة للعلاقة «يســند إلى» (Is_assigned) حتى يتم ربط المريض بسرير معين فى المستشفى.

شكل رقم (٣-٥): تمثيل التخصيص الكامل فى علاقات الأنواع الرئيسية والأنواع الفرعية

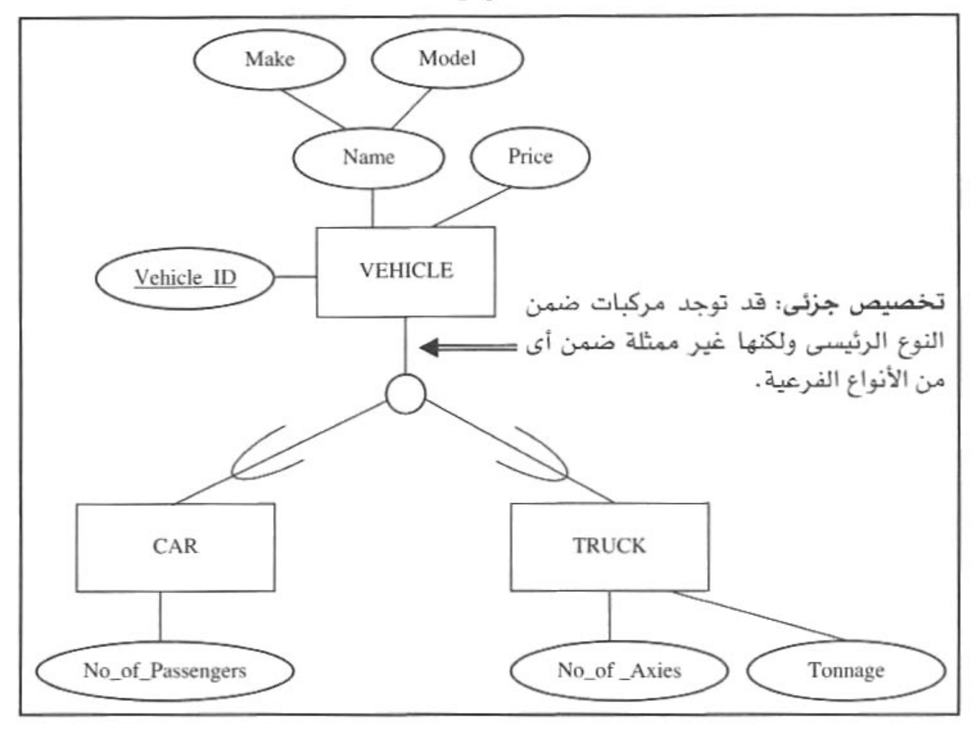


٣-١-١-٢-١-٢ التخصيص الجزئى (Partial Specialization):

يوضح الشكل رقم (٣-٦) مثالاً لنوع رئيسى اسمه «مركبة» (VEHICLE) ونوعين فرعيين يرتبطان به هما: «سيارة» (CAR) وشاحنة (TRUCK). وكما أوضحنا أعلاه فإن الخصائص المشتركة لكافة المركبات (بغض النظر عن أنواعها الفرعية) قد تم ربطها بالنوع الرئيسى. فرقم لوحة المركبة (Vehicle_ID)، واسمها (Name) (الذى يتكون من صنعها (Make) مثل تويوتا كورولا، وسنة الصنع (Model))، وسعرها (Price)، هى خصائص مشتركة لكافة أنواع المركبات. أما الخصائص المتعلقة بنوع فرعى معين مثل خاصية «عدد الركاب» (No_of_Passengers)، وخاصية «عدد محاور الدفع» (No_of_Axies)، وخاصية «الحمولة بالطن» (Tonnage)، فقد تم ربطها بالنوع الفرعى نفسه. ولنفترض فى هذا المثال وجود أنواع أخرى من المركبات التى يأتى من ضمنها «الدراجات النارية» (MOTORCYCLE). ولأن هذا النوع من المركبات لا يوجد له نوع فرعى خاص به بمعنى أن الخصائص المشتركة المرتبطة بالنوع الرئيسى تكفى لتمثيله دون وجود خصائص تتعلق به فقط دون غيره من المركبات، فإن قيد التخصيص فى هذه الحالة يعد تخصيصاً جزئياً. فعند إضافة مركبة (VEHICLE) من فئة سيارة ضمن حالات النوع الرئيسى لا بد من إضافة ما يقابلها ضمن حالات النوع الفرعى «سيارة» (CAR)، وعند إضافة مركبة جديدة (VEHICLE) من فئة شاحنة ضمن النوع

الرئيســـى لا بد من إضافة ما يقابلها ضمن حالات النوع الفرعى «شاحنة» (TRUCK)، أما عند إضافة مركبة (VEHICLE) من فئة «دراجة نارية» فإنه يكتفى بإضافتها ضمن حـــالات النوع الرئيســـى فقط. ويمثــل التخصيص الجزئى بخط منفـــرد يصل النوع الرئيسى وهو «مركبة» (VEHICLE) بنقطة تفرع الأنواع الفرعية (وهى الدائرة).

شكل رقم (٣-٦): تمثيل التخصيص الجزئى فى علاقات الأنواع الرئيسية والأنواع الفرعية



٣-١-١-٢-٢ قيد الانفصال (Disjointness Constraint):

إن قيد التخصيص الذى سبق شرحه أعلاه يتحدث عن العلاقة الرأسية بين النوع الرئيســـى وأنواعه الفرعية، بمعنى أنه يقيد حالات النوع الرئيســـى من حيث ضرورة وجـــود كل حالة ضمن أحد الأنواع الفرعية التى ترتبط به من عدم وجودها. أما قيد الانفصـــال (Disjointness Constraint) فيتحدث عن العلاقة الأفقية بين الأنواع الفرعية

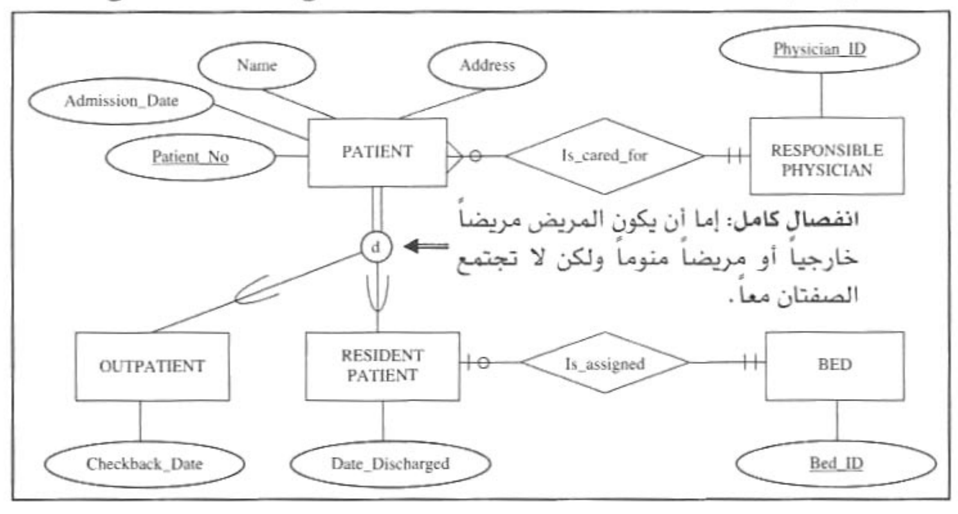
التى تتبع لنوع رئيســى معين، بمعنى أنه يقيد الأنواع الفرعية من حيث إمكانية وجود حالــة معينــة فى أكثر من نــوع فرعى واحد فــى نفس الوقت. وكمــا هو الحال فى قيــد التخصيص، يوجد قاعدتان لقيد الانفصال وهمــا الانفصال الكامل والانفصال المتداخــل. فالانفصال الكامل يعنى أن وجود حالة ما من حالات النوع الرئيســى فى نوع فرعى معين لا يمكن أن توجد ضمن حالات أنواع فرعية أخرى ترتبط بنفس النوع الرئيســى. أما الانفصال المتداخل فيعنى أن وجود حالة ما من حالات النوع الرئيسى فــى نوع فرعى مــا من الممكن أن توجد فــى أنواع فرعية أخرى ترتبــط بنفس النوع الرئيسي. وفيما يلى إيضاح لكلتا القاعدتين.

٣-١-١-٢-٢-١ الانفصال الكامل (Total Disjoint):

يبين الشــكل رقم (٣-٧) المثال المتعلق بمرضى أحد المستشــفيات بعد إيضاح رمز الانفصال عليه. إن قاعدة العمل المعمول بها فى هذا المثال هى كما يلى: إن أى مريض فى المستشــفى إما أن يكون مريضاً منومــاً أو مريضاً فى العيادات الخارجية ولكنه لا يمكــن أن يكون مريضاً منوماً ومريضاً فى العيادات الخارجية فى وقت واحد. وتعنى قاعــدة العمل هذه أن القيد بــين الأنواع الفرعية هو انفصال كامل؛ إذ إنه لا يمكن أن يوجــد أى مريــض ضمن حالات أكثر من نوع فرعى واحد فى وقت واحد. ويمثل قيد الانفصال الكامل بالحرف (d)، الذى يمثل الحرف الأول من كلمة «انفصال» (Disjoint)، داخل دائرة التفرع.

ويلاحظ فى هذا التمثيل أن حالة أى مريض من الممكن أن تكون فى أى نوع فرعى، ولكنهــا لا يمكن أن توجَد فى أكثر من نوع فرعى فى نفس الوقت. فالمريض المنوم قد يخرج من المستشــفى ويصبح مراجعاً للعيادات الخارجية وكذلك هو الحال بالنســبة لمريض العيادات الخارجية الذى قد يصبح مريضاً منوماً فى وقت آخر. ويمكن قراءة القيدين الممثلين فى الشــكل (قيــد التخصيص وقيد الانفصــال) مجتمعين كما يلى: يوجد تخصيص كامل بين النوع الرئيسى والأنواع الفرعية المرتبطة به لكون أى مريض فى المستشــفى لا بد أن يمثل ضمن أحد الأنواع الفرعية، ويوجد أيضاً انفصال كامل لكون أى مريض لا يمكن أن يكون موجوداً فى أكثر من نوع فرعى واحد فى أى وقت.

شكل رقم (٣-٧): تمثيل الانفصال الكامل فى علاقات الأنواع الرئيسية والأنواع الفرعية



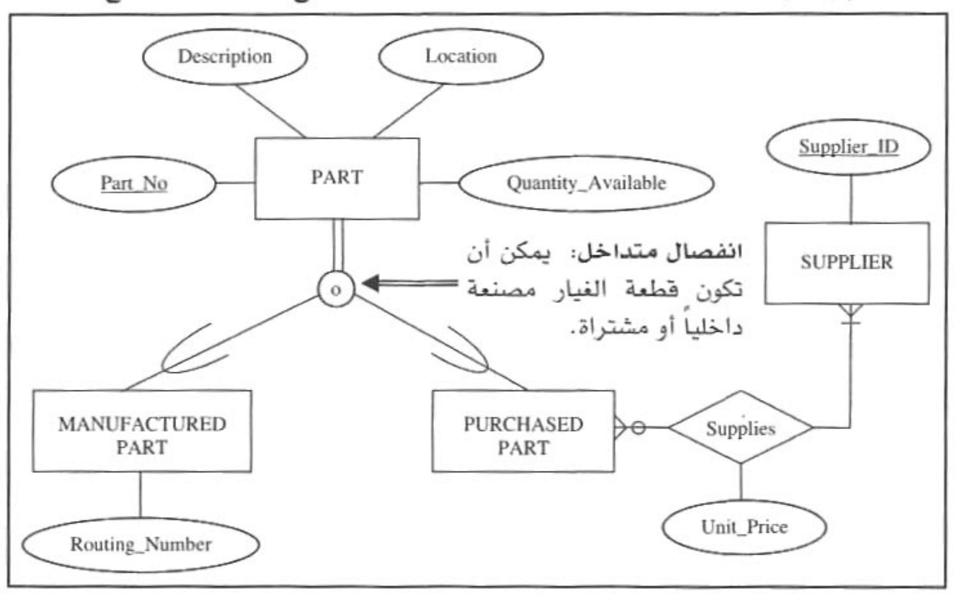
٣-١-١-٢-٢-٢ الانفصال المتداخل (Overlapping Disjoint):

يوضح الشكل رقم (٣-٨) فئة كينونة «قطعة غيار» (PART) يرتبط بها نوعان فرعيان: النوع الأول منهما يمثل قطع الغيار المصنعة داخلياً (فى نفس المنظمة) بمسمى (MANUFACTURED_PART)، والنوع الثانى يمثل قطع غيار مشتراة (PURCHASED_PART). فى هذا المثال تم ربط الخصائص المشتركة لقطع الغيار بالنوع الرئيسى. وهذه الخصائص هى: رقم قطعة الغيار (Part_No) الذى يعد معرفاً لقطع الغيار، ووصف لقطعة الغيار (Description)، ومكان وجود قطعة الغيار (Location)، والعدد المتوافر منها (Quantity_Available). فعلى سبيل المثال، يتوافر من قطعة الغيار رقم (٥٠٠)، وهى مصابيح خلفية يسرى لسيارة فورد توراس ٢٠٠٤، (١٥) قطعة موجودة فى الرف رقم (٤) من الممر رقم (٥) فى مستودع المنظمة. ولنفترض أنه تم تصنيع (١٠) قطع من قطعة الغيار هذه محلياً، فى حين تم توريد (٥) قطع من مورد ما. فى هذه الحالة تمثل قطعة الغيار هذه ضمن حالات النوع الفرعى (MANUFACTURED_PART) وفى نفس الوقت ضمن حالات النوع الفرعى (PURCHASED_PART). ويعنى هذا التمثيل أن قيد الانفصال متداخل لكون قطع الغيار قد توجد فى أكثر من نوع فرعى واحد فى وقت واحد. ويلاحظ فى هذا التمثيل أن النوع الفرعى المتعلق بالقطع المصنعة محلياً له خاصية تميزه عن النوع الفرعى المتعلق بقطع الغيار المشتراة من

مورد ما، وأن قطع الغيار المشـــتراة ترتبط بعلاقة مع كينونة «المورد» (SUPPLIER) التى لا يرتبط فيها النوع الفرعى الآخر. كما يلاحظ فى هذا التمثيل أنه لا يتتبع كل قطعة علـــى حدة ولكنه يتتبـــع مجموعة من القطع من نفس النوع ولها نفس الرقم كمجموعة واحدة. أما إذا أردنا تمثيل كل قطعة على حدة فإنه يجب إدخال الرقم التسلسلى لكل قطعة (Serial Number) وتكون الكمية المتوافرة منها إما (١) أو (٠) حسب وجودها فى المستودع، مع الإبقاء على بقية مكونات التمثيل فى النموذج كما هى.

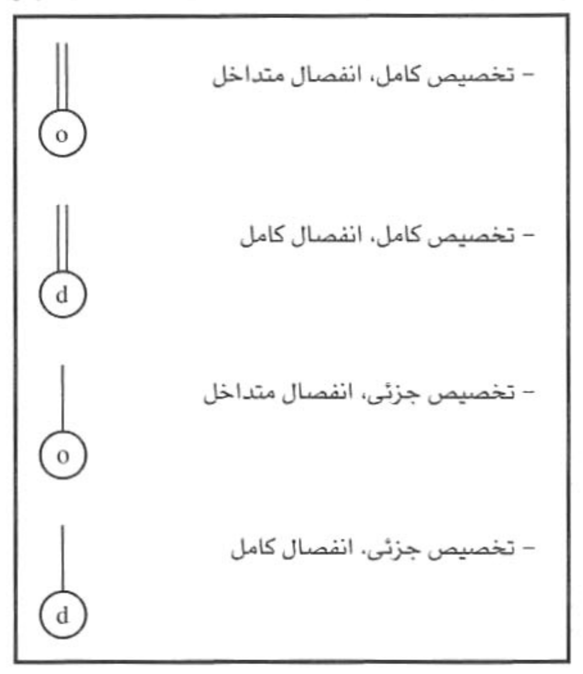
ويمثـــل قيد الانفصال المتداخـــل بالحرف (o)، الذى يمثل الحـــرف الأول من كلمة «متداخـــل» (Overlap)، داخل دائرة التفرع. ويمكن قراءة القيدين الممثلين فى الشـــكل (قيد التخصيص وقيد الانفصال) مجتمعين كما يلى: يوجد تخصيص كامل بين النوع الرئيســـى والأنواع الفرعية المرتبطة به بمعنى أن أية قطعة غيار يجب أن توجد ضمن أحد الأنواع الفرعية، ويوجد انفصال متداخل لكون بعض قطع الغيار قد توجد ضمن قطع الغيار المشتراة وفى الوقت نفسه ضمن قطع الغيار المصنعة محلياً.

شكل رقم (٣-٨): تمثيل الانفصال المتداخل فى علاقات الأنواع الرئيسية والأنواع الفرعية



يحتوى الشـــكل رقم (٣-٩) على كافة التوليفات الأربعة المحتملة للقيدين الســـابقين (قيد التخصيص وقيد الانفصال) التى يمكن استخدام المناسب منها حسب قواعد العمل المعمول بها فى المنظمة، وذلك عند استخدام علاقة النوع الرئيسى والأنواع الفرعية.

شكل رقم (٣-٩): التوليفات الأربعة المحتملة لقيد التخصيص وقيد الانفصال



٣-١-١-٣ تعريف مميز للأنواع الفرعية (Defining Subtype Identifiers):

عند إضافة حالة جديدة ضمن حالات نوع رئيسى معين فإننا نحتاج إلى معرفة النوع الفرعى (أو الأنواع الفرعية) التى يجب إضافة هذه الحالة إليها إذا كان لا بد من إضافة الحالة إلى أحد الأنواع الفرعية. ولأننا قد سبق أن شرحنا قواعد الإضافة لكل من قيد التخصيص وقيد الانفصال، فإننا نقدم هنا طريقة مبسطة تعتمد على مميز للأنواع الفرعية لتطبيق قواعد الإضافة هذه (Hoffer et al, 2002). ومميز الأنواع الفرعية ما هو إلا خاصية تتم إضافتها للنوع الرئيسى بحيث تحدد النوع الفرعى (أو الأنواع الفرعية) التى يجب إضافة الحالة إليها.

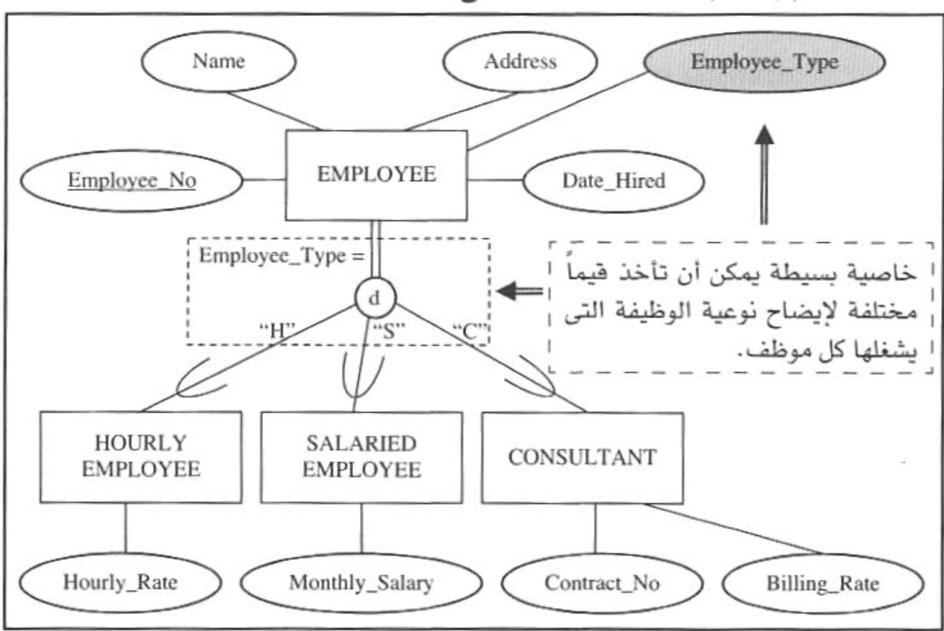
٣-١-١-٣-١ الانفصال الكامل (Total Disjoint):

يمثل الشكل رقم (٣-١٠) علاقة النوع الرئيسى «موظف» بالأنواع الفرعية الثلاثة التى ترتبط به والتى سبق شرحها أعلاه حيث يوجد قيد تخصيص كامل بين النوع الرئيسى

والأنواع الفرعية، بمعنى أن أى موظف موجود ضمن حالات النوع الرئيسى لا بد أن توجد لــه حالة فى أحد الأنــواع الفرعية. كما يوجد قيد انفصال كامــل بين الأنواع الفرعية، بمعنى أن أى موظف لا يمكن أن يوجد له حالة فى أكثر من نوع فرعى واحد.

ويوضح الشكل رقم (٣-١٠) مميز الأنواع الفرعية الذى حدد باللون الداكن بمسمى «نــوع الموظف» (Employee_Type). وعند إضافة موظف جديد فإنه يجب إدخال قيمة لهذا المميز بحيث تكون قيمته إما (H) للدلالة على أن الموظف يعمل بأجر الساعات، أو (C) للدلالة على أن الموظف يعمل مستشاراً وفق عقد استشارى، أو (S) للدلالة على أن الموظف يعمل بالأجر الشــهرى. وبناء على القيمة المدخلة لهذه الخاصية، تتم إضافة حالة للموظف الجديد ضمن حالات النوع الفرعى المناســب. كما يســتخدم فى هذا التمثيل اســم المميز بين النوع الرئيســى ونقطة التفرع كشرط يجب أن تساوى قيمته إحــدى القيم المدونة على الخطوط الواصلة بين نقطة التفرع والأنواع الفرعية. فعلى ســبيل المثال، عند إضافة موظف قيمة خاصية «نــوع الموظف» (Employee_Type) هى «مستشــار» (C) فإن هذا يعنى أن شرط الإضافة سيكون (Employee_Type = "C") مما يؤدى إلى إضافة حالة لهذا الموظف ضمن النوع الفرعى (CONSULTANT).

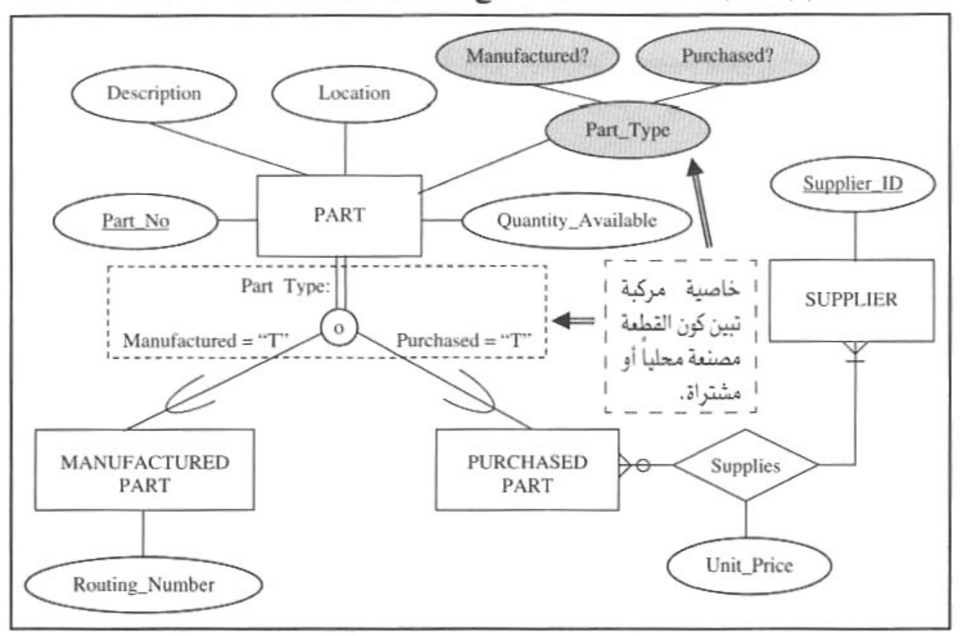
شكل رقم (٣-١٠): تعريف مميز الأنواع الفرعية فى حالة الانفصال الكامل



٣-١-١-٣-٢ الانفصال المتداخل (Overlapping Disjoint):

يمثل الشـــكل رقم (٣-١١) علاقة النوع الرئيسى «قطعة غيار» بالنوعين الفرعيين اللذين يرتبطان به، وقد ســبق شـــرحها أعلاه، حيث يوجد قيـد تخصيص كامل بين النوع الرئيسى والأنواع الفرعية، بمعنى أن أى قطعة غيار موجودة ضمن حالات النوع الرئيســى لا بد أن توجد لها حالة فى أحد الأنــواع الفرعية. كما يوجد قيد انفصال متداخل بــين الأنواع الفرعية، بمعنى أن قطعة الغيار قـد توجد بوصفها حالة ضمن قطع الغيار المصنعة محلياً وفى الوقت نفسه حالة ضمن قطع الغيار المشتراة.

شكل رقم (٣-١١): تعريف مميز الأنواع الفرعية فى حالة الانفصال المتداخل



ويوضح الشكل رقم (٣-١١) مميز الأنواع الفرعية الذى حدد باللون الداكن بمسمى «نوع قطعة الغيار» (Part_Type) والذى يختلف بشـــكل بسيط عن التمثيل السابق، حيث مثل المميز كخاصية مركبة عوضاً عن تمثيله كخاصية بسيطة. وبهذه الطريقة نستطيع تمثيل التوليفات المناســبة لكل قطعة غيار من حيث كونها مصنعة محلياً أو مشـــتراة. وتأخــذ كل خاصية بســيطة من مكونات الخاصية المركبة قيمـــة منطقية واحدة: إما صح (True) أو خطأ (False). فعندما يتم إدخال قطعة غيار جديدة ضمن حالات النوع الرئيسى وتكون القطعة مصنعة محلياً ومشتراة فى نفس الوقت تكون قيم الخاصيتين البســيطتين كما يلى: (Manufactured = True, Purchased =True). أما إذا كانت القطعة

مصنعــة محلياً فقط فتكون قيم الخاصيتين البســيطتين كما يلــى: (Manufactured = True, Purchased = False). أما إذا كانت القطعة مشــتراة فقط فتكون قيم الخاصيتين البسيطتين كما يلى: (Manufactured = False, Purchased = True).

٣-١-١-٤ التعميم والتخصيص (Generalization and Specialization):

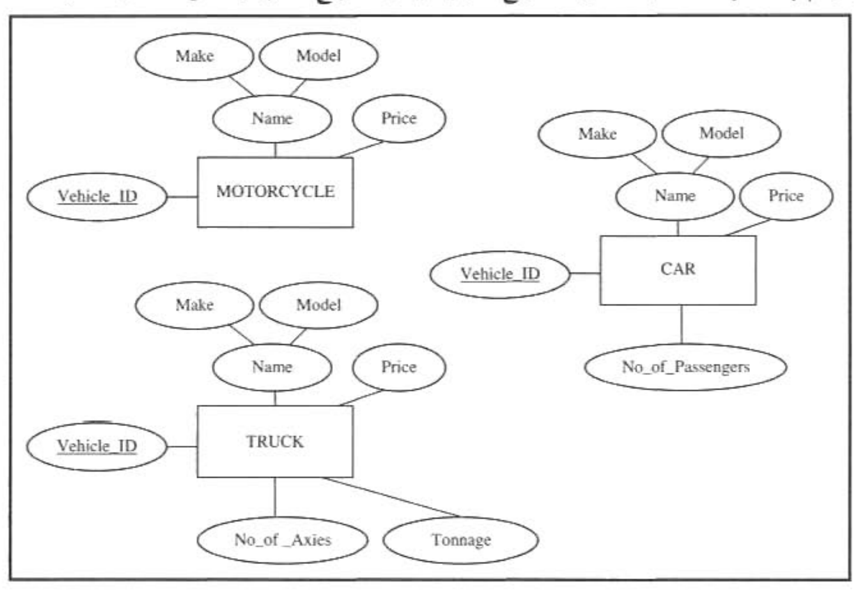
لا شــك أن علاقة الأنواع الرئيســية والأنواع الفرعية فى نموذج البيانات كينونة - علاقة المطور مبدأ جيد يمكننا من وصف العلاقات بين الكينونات الموجودة فى المنظمة بشــكل أكثر دقة. إلا أن البيانات تختلف من منظمة إلى أخرى، وقد يســتعصى الأمر على الشــخص الذى ســيقوم بنمذجة البيانات التعرف على الكينونات التى يمكن أن ترتبط بعلاقة نوع رئيســى وأنواع فرعية منذ البدء فى تصميم نموذج البيانات. لهذا يمكن استخدام عملية التعميم أو التخصيص لاستكشاف مثل هذه العلاقات.

٣-١-١-٤-١ التعميم (Generalization):

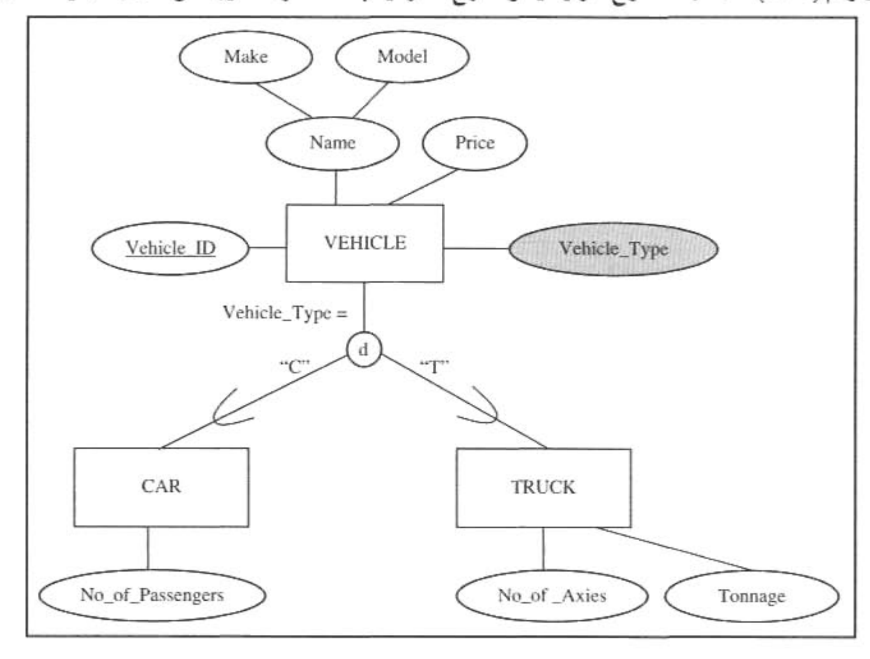
إن عملية التعميم فى نمذجة البيانات هى عملية تتم من الأسفل إلى الأعلى بمعنى أنهــا عمليــة تعريف لفئة كينونة أعم مــن مجموعة من فئــات الكينونات المخصصة. فعلى سبيل المثال، يوضح الشكل رقم (٣-١٢) ثلاثة أنواع من الكينونات هى: السيارة (CAR)، والشــاحنة (TRUCK)، والدراجة النارية (MOTORCYCLE). وعلى الرغم من أنــه يمكن تمثيل هذه الكينونــات الثلاث ضمن نموذج البيانــات كينونة - علاقة كما هى، إلا أنه بنظرة فاحصة يتضح أن الكينونات الثلاث لديها خصائص مشتركة وهى: الاسم، والسعر، ورقم المركبة. ويقترح مثل هذا التشابه فى بعض الخصائص بين فئات الكينونات المختلفة على مُنمذج البيانات التفكير فى إمكانية تعريف فئة كينونة أعم.

ونظــراً لهذا التشــابه فإنــه يمكن تعريف فئــة كينونة أعم وهى كينونــة «المركبة» (VEHICLE). ويوضــح الشــكل رقم (٣-١٣) فئة الكينونــة الجديدة بالإضافة لعلاقة النوع الرئيسى والأنواع الفرعية. ويلاحظ فى الشكل نفسه وجود نوعين فرعيين فقط عوضاً عن ثلاثة؛ وذلك لكون كينونة السيارة وكينونة الشاحنة لهما خصائص تميزهما عن بعضهما، وهذه الخصائص غير موجودة ضمن الخصائص العامة لكافة الكينونات والمرتبطة بالنوع الرئيسى، أما فئة كينونة الدراجة النارية فجميع خصائصها خصائص عامــة ومن ثم لا داعى لتمثيلها بوصفها نوعاً فرعياً. كما يتم إضافة القيود بين النوع الرئيسى والأنواع الفرعية، وهى فى هذا المثال تخصيص جزئى لكون الدراجات النارية لا تظهــر ضمــن أى من الأنواع الفرعيــة، وانفصال كامــل لأن أى مركبة لا يمكن أن تكون ضمن حالات فئة كينونة الســيارة وفى الوقت نفســه ضمن فئة كينونة الشاحنة. بالإضافة لذلك، يتم تعريف خاصية المميز للأنواع الفرعية وربطها بالنوع الرئيسى.

شكل رقم (٣-١٢): التعرف على الأنواع الرئيسية والأنواع الفرعية من خلال عملية التعميم



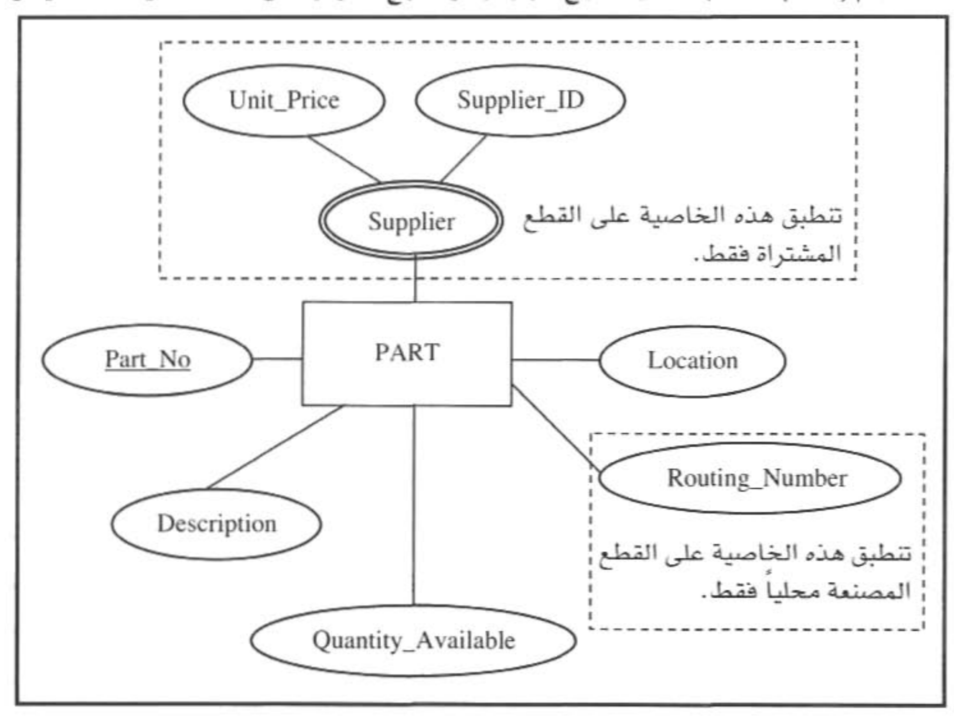
شكل رقم (٣-١٣): نمذجة الأنواع الرئيسية والأنواع الفرعية بعد التعرف عليها من خلال عملية التعميم



٣-١-١-٤-٢ التخصيص (Specialization):

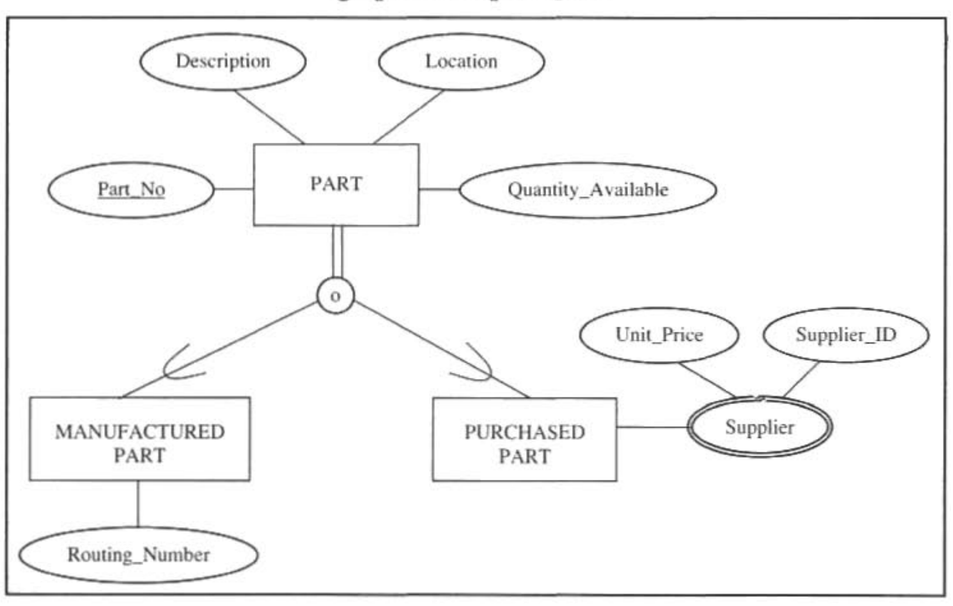
إن عمليـــة التخصيص هـى عكس عملية التعميم حيث تتم من الأعلى إلى الأســـفل بمعنى أنها عملية تعريف لفئات كينونات مخصصة من فئة كينونة أعم. فعلى ســبيل المثال، يوضح الشـــكل رقم (٣-١٤) كينونة «قطعة غيـــار» (PART) التى يمكن أن تكون مشـــتراة أو مصنعة محلياً (أو كلتيهما معاً فى وقت واحد). ويلاحظ فى هذا التمثيل وجـــود خاصية متعددة القيم وهـى خاصية «المورد» (Supplier) التى تتكون من الخاصية البســيطة «رقم المورد» (Supplier_ID) وسعر القطعة، مع ملاحظة أن سعر القطعة قد يتغيـــر من مورد إلى آخـــر. وترتبط هذه الخاصية بقطع الغيار المشـــتراة فقط. كما يلاحـــظ فى هذا التمثيل وجود الخاصية البســيطة «رقم المصدر» (Routing_Number) التى ترتبط بالقطع المصنعة محلياً فقط. وفى حالة كون القطع مصنعة محلياً ومشتراة فى نفس الوقت، فســيكون للقطعة قيم ضمن كلتا الخاصيتين (المورد، ورقم المصدر). ولكـــون خصائص قطع الغيار قد تختلف حســـب طبيعة كون قطعة الغيار مشـــتراة أو مصنعة محليـــاً، فإن مثل هذا التمثيل يقترح على منمذج البيانات التفكير فى إمكانية تعريف فئات خاصة لكل نوع من أنواع قطع الغيار.

شكل رقم (٣-١٤): التعرف على الأنواع الرئيسية والأنواع الفرعية من خلال عملية التخصيص



ونظــراً لوجود اختلاف فى بعض خصائص كلا النوعين من أنواع قطع الغيار فإنه يمكن تعريف فئات كينونــات فرعية مخصصة، وهى: كينونة «القطع المصنعة محلياً» (MANUFACTURED_PART)، وكينونــة «القطــع المشــتراة» (PURCHASED_PART). وترتبــط كلتا الكينونتين بالنوع الرئيســى مــن خلال علاقة النوع الرئيســى والأنواع الفرعيــة، كما يرتبط كل نــوع فرعى بالخصائص الخاصة به فــى حين يرتبط النوع الرئيسى بالخصائص العامة لكافة أنواع الكينونات. ونظراً لكون كل قطعة غيار يجب أن تكون إما مصنعة محلياً أو مشــتراة، فإن هذا يعنى وضع قيد تخصيص كامل. أما قيــد الانفصال فهو انفصال متداخــل؛ وذلك لكون بعض قطع الغيار قد تكون مصنعة محلياً ومشتراة فى وقت واحد؛ مما يعنى أن بعض الحالات الموجودة فى النوع الرئيسى قد توجد فى كلا النوعين الفرعيين. وبناءً على ذلك يصبح النموذج الســابق كما هو موضح فى الشكل رقم (٣-١٥).

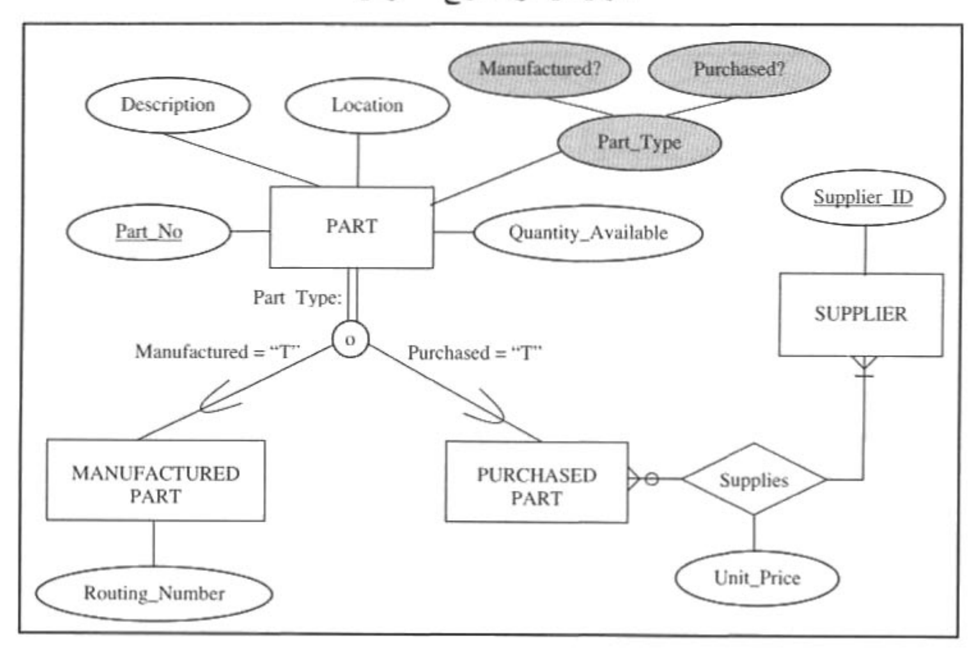
شكل رقم (٣-١٥): نمذجة الأنواع الرئيسية والأنواع الفرعية بعد التعرف عليها من خلال عملية التخصيص



وبملاحظــة كون الخاصية المرتبطة بالقطع المشــتراة أنها خاصيــة متعددة القيم وفى الوقت نفســه تحتوى على قيمة بســيطة تعد مميزاً للخاصية وهى «رقم المورد»

(Supplier_ID)، فإن هذه الخاصية يمكن تحويلها إلى كينونة قائمة بذاتها ترتبط بالنوع الفرعى من خلال علاقة «يورد» (Supplies). كما ترتبط بالعلاقة نفسها خاصية «سعر الوحدة» (Unit_Price)؛ وذلك لأن سعر الوحدة ليست خاصية من خصائص المورد، كما أنها ليست خاصية من خصائص قطعة الغيار، وإنما هى خاصية للعلاقة نفسها بمعنى أن قيمــة هذه الخاصية تختلف باختلاف المورد واختلاف قطعة الغيار. فعلى ســبيل المثال، قد يكون لإحدى قطع الغيار المشتراة أكثر من سعر للوحدة حسب الموردين الذين قاموا بتوريد القطعة. لذلك لا يمكن ربط هذه الخاصية بأى من الكينونتين، وإنما يتم ربطها بالعلاقة التى تربط الكينونتين ببعضهما. وبذلك يصبح الشــكل النهائى لعملية التخصيص كما هو موضح فى الشــكل رقم (٣-١٦)، مع ملاحظة إضافة مميز الأنواع الفرعية «نوع القطعة» (Part_Type) ضمن خصائص النوع الرئيسى.

شكل رقم (٣-١٦): تحديد العلاقات الخاصة بالأنواع الفرعية فى علاقات الأنواع الرئيسية والأنواع الفرعية



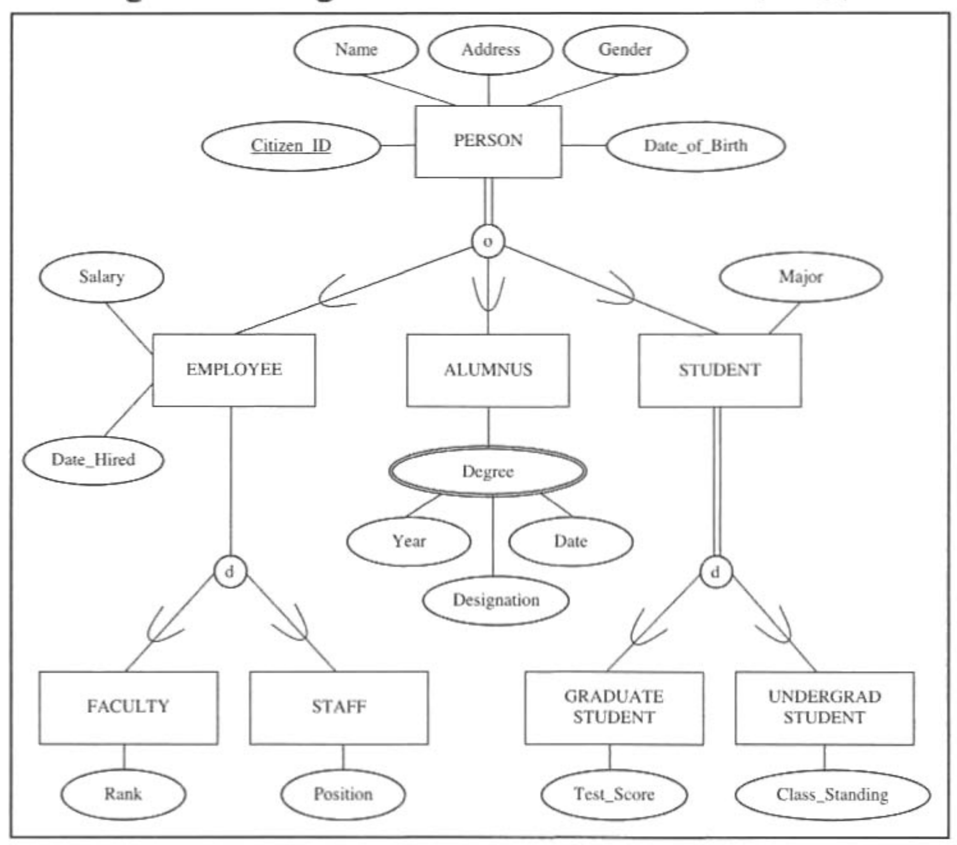
مما ســبق تتضح أهمية عمليتى التعميم والتخصيص؛ إذ إنهما تمكنان من تفحص النموذج المبدئى للبيانات ومن ثم إمكانية التعرف على الأنواع الرئيسية والأنواع الفرعية؛

مما يســهم فى تطوير النموذج بحيث يعكس علاقات البيانات والقيود المفروضة عليها بشكل أكثر دقة.

٣-١-١-٥ هرميات الأنواع الرئيسية والأنواع الفرعية (Supertype/Subtype Hierarchies):

عنــد وجود علاقة نوع رئيســى بأنواع فرعية فإنه من الممكــن أن يرتبط كل نوع من الأنــواع الفرعية بأنواع فرعية خاصة به، وبذلك تتشــكل هرمية من الأنواع الرئيســية والأنــواع الفرعية، بحيــث يصبح النوع الفرعى الذى ترتبط بــه أنواع فرعية خاصة به نوعاً رئيســياً للأنواع الفرعية المرتبطة به (Elmasri and Navathe, 2004). ويوضح الشكل رقم (٣-١٧) أحد الأمثلة الشائعة الذى يمثل بيانات الموارد البشرية فى إحدى الجامعات والعلاقات فيما بينها، كما تم إيضاح المفاهيم والرموز التى تم شرحها حتى الآن عليه.

شكل رقم (٣-١٧): أحد الأمثلة الشائعة لهرميات الأنواع الفرعية والأنواع الرئيسية



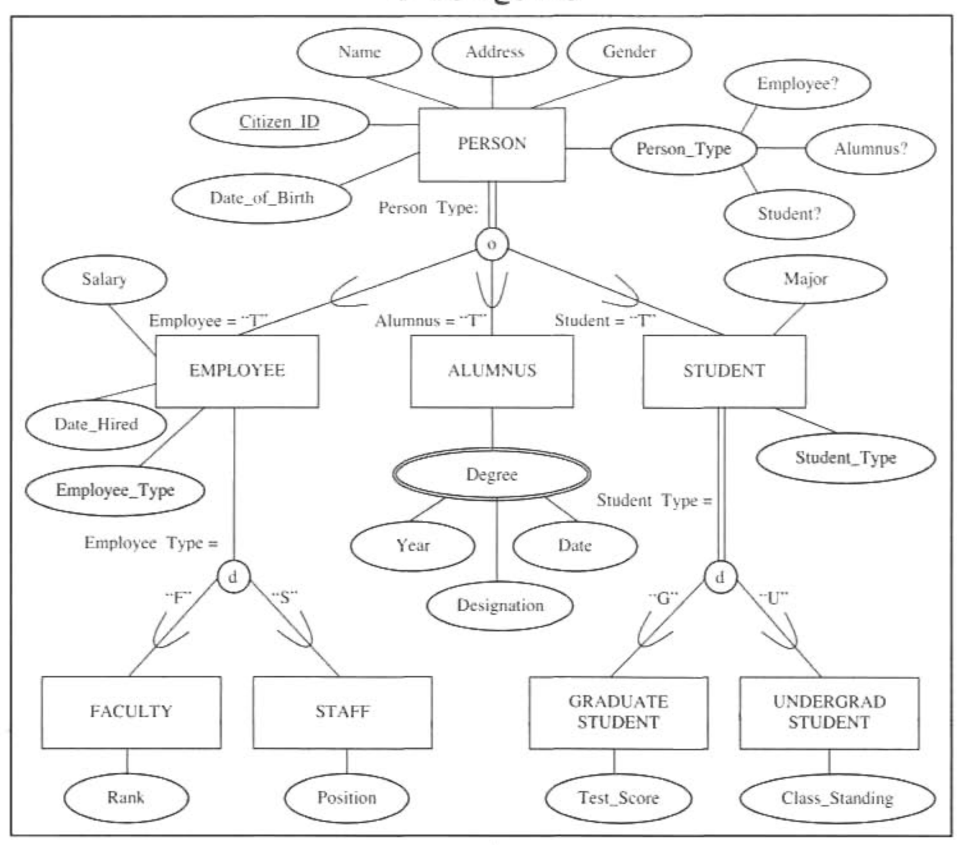
لقد تم فى الشكل رقم (٣-١٧) تعريف كينونة نوع رئيسى وهى كينونة «الأشخاص» (PERSON) التى ترتبط فيها كافة الخصائص المشتركة للأشخاص التابعين للجامعة من موظفين وطلبة وخريجين. وهذه الخصائص هى: رقم «السجل المدنى» (Citizen_ID) الذى يمثل معرف الكينونة، و«الاسم» (Name)، و«العنوان» (Address)، و«الجنس» (Gender)، و«تاريخ الميلاد» (Date_of_Birth). بعد ذلك تم تخصيص ثلاثة أنواع فرعية هى: فئة «الموظفين» (EMPLOYEE)، وفئة «الخريجين» (ALUMNUS)، وفئة «الطلبة» (STUDENT). كما تم ربط كل نوع فرعى بالخصائص المتعلقة به فقط حيث تم ربط النوع الفرعى الذى يمثل الموظفين بخاصية «الراتب» (Salary) وخاصية «تاريخ التعيين» (Date_Hired) التى تمثل خصائص عامة لكافة فئات الموظفين. كما تم ربط النوع الفرعى الذى يمثل الخريجين بخاصية متعددة القيم بمسمى «الدرجة العلمية» (Degree) وذلك لكون الخريج قد يكون حاصلاً على أكثر من درجة علمية من نفس الجامعة. كما تم توصيف الدرجة العلمية المتعددة القيم على أساس أنها مركبة أيضاً حيث تتكون من الخصائص البسيطة التالية: خاصية «سنة الحصول على الدرجة العلمية» (Year)، وخاصية «تاريخ الحصول على الدرجة العلمية» (Date)، وخاصية «نوع الدرجة العلمية» (Designation). أما بالنسبة للنوع الفرعى الذى يمثل الطلبة فقد تم ربطه بخاصية «التخصص» (Major). ونظراً لكون أى شخص فى الجامعة يجب أن يكون من ضمن أحد الأنواع الفرعية الثلاثة، فقد تم وضع قيد تخصيص كامل. أما قيد الانفصال فقد تم وضعه على أساس أنه انفصال متداخل؛ وذلك لكون الخريج قد يكون موظفاً فى الجامعة أيضاً، كما أن الموظف قد يكون طالباً فى الجامعة أيضاً.

بعد ذلك تم النظر فى كل نوع فرعى على حدة لمعرفة إمكانية تخصيص أنواع فرعية منه، إذ لوحظ أن الموظف قد يكون «عضواً لهيئة التدريس» (FACULTY) أو أحد أفراد «الطاقم المساعد فى عملية التدريس» (بما فى ذلك من إداريين وسكرتارية (STAFF)). وبناء على ذلك تم تخصيص فئتين من النوع الفرعى «موظف» وتم ربط كل نوع فرعى بالخصائص التى ترتبط به فقط. ويلاحظ أن قيد التخصيص هو تخصيص جزئى مما يعنى وجود موظفين آخرين لا ينتمون لأى من النوعين الذين تم تخصيصهما، كما يلاحظ أن قيد الانفصال هو انفصال كامل مما يعنى أن عضو هيئة التدريس لا يمكن أن يكون من ضمن الطاقم المساعد فى العملية التدريسية أو العكس. بعد ذلك تم النظر فى النوع الفرعى الذى يمثل الطلبة حيث لوحظ وجود فئتين من الطلبة وهما: فئة «طلبة الدراسات العليا» (GRADUATE_STUDENT) وفئة طلبة «دراسات البكالوريوس» (UNDERGRADUATE_STUDENT) وتم ربط كل فئة بالخصائص التى

ترتبط بها فقط. ولأن أى طالب لا بد أن يكون إما طالباً فى الدراسات العليا أو طالباً فـــى مرحلة البكالوريوس فإنه قد تم تمثيـــل قيد التخصيص على أنه تخصيص كامل. كمـــا تم تمثيل قيد الانفصال على أنـــه تخصيص كامل لكون الطالب لابد أن يكون إما طالبـــاً فى مرحلـــة البكالوريوس أو طالباً فى مرحلة الدراســـات العليا ولكن ليس فى المرحلتين فى نفس الوقت.

تجدر الملاحظة هنا أن أى نوع فرعى فى هرمية الأنواع الرئيسية والأنواع الفرعية يـــرث كافة الخصائص والعلاقات للأنـــواع التى تعلوه فى الهرميـــة. كما يتم إضافة خاصية مميز الأنواع الفرعية لكل كينونة يتم تخصيصها وليس فقط للنوع الرئيســـى الذى فى أعلى الهرمية كما يوضح الشكل رقم (٣-١٨).

شكل رقم (٣-١٨): إضافة خاصية مميز الأنواع الفرعية فى هرميات الأنواع الفرعية والأنواع الرئيسية

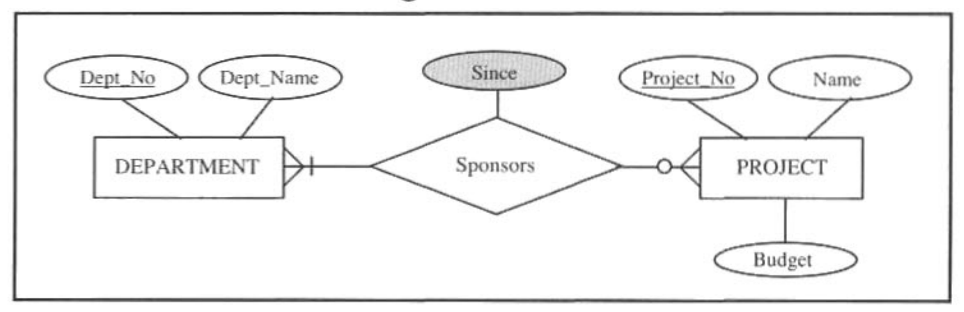


ويوضح المثال السابق أيضاً منهجية التخصيص، حيث تم البدء فى هذا المثال بالنوع العام وهو «الأشخاص» وتم تخصيصه شيئاً فشيئاً حسب الخصائص التى تميز كل نوع فرعى عن بقيـــة الأنواع الفرعية حتى الوصول للهرمية الكاملة التى تمثل كافة الموارد البشرية فى الجامعة.

٣-١-١-٦ التجميع (Aggregation):

عُرِّفــت فئة العلاقة بأنها ارتباط بين نوعين أو أكثر من فئات الكينونات. إلا أنه فى بعــض الأحيان نحتاج إلى نمذجة فئة علاقة بين فئة كينونـــة ما، من جانب، ومجموعة مــن الكينونات والعلاقــات مجتمعة مع بعضها، مــن جانب آخر. فعلى ســبيل المثال، لنفترض وجود فئة كينونة بمســـمى «مشـــروع» (PROJECT) وفئة كينونة بمسمى «قسم» (DEPARTMENT) وأن كل مشـــروع «مدعوم مالياً» (Sponsored) من خلال قسم واحد أو أكثر، وأن القسم الواحد يمكن أن لا يدعم مالياً أى مشروع أو أنه يدعم أكثر من مشروع. وعندما يبدأ قسم بدعم مشروع ما، يوجد هناك تاريخ لبداية الدعم. يمكن نمذجة هذا الوضع من خلال علاقة «يدعم مالياً»، كما هو موضح فى الشكل رقم (٣-١٩).

شكل رقم (٣-١٩): علاقة «الدعم المالى» التى تربط بين فئة كينونة الأقسام وفئة كينونة المشاريع



لنفترض الآن أنه كلما قام قسم بدعم مشروع ما، يقوم بتكليف موظف (أو مجموعة موظفين) «لمتابعة ســـير المشـــروع» (Monitors). من المنطقى فى هذه الحالة أن علاقة «المتابعة» (Monitors) تربط بين فئة كينونة الموظفين، من جانب، وعلاقة «الدعم المالى» (Sponsors)، وليس فئة كينونة المشروع أو فئة كينونة القسم. لهذا السبب يمكن استخدام «التجميع» (Aggregation) الذى يمكننـــا من تمثيل مثل هذا الوضع (Ramakrishnan and Gehrke, 2003). ويُمكــن التجميع من توضيح أن فئة علاقة (عوضاً عن كينونة) ترتبط

بفئة كينونة من خلال فئة علاقة أخرى. وتمثل فئة العلاقة المجمعة من خلال وضعها داخل مســتطيل منقط الأضلاع. ويوضح الشــكل رقم (٣-٢٠) علاقة «الدعم المادى» والكينونــات التى تربط بينها بعد تجميعها (من خلال وضعها داخل مســتطيل منقط الأضلاع) للدلالة على أنها تدخل مجتمعة فى علاقة أخرى هى علاقة «المتابعة».

شكل رقم (٣-٢٠): علاقة «متابعة سير المشروع» عند استخدام مفهوم التجميع

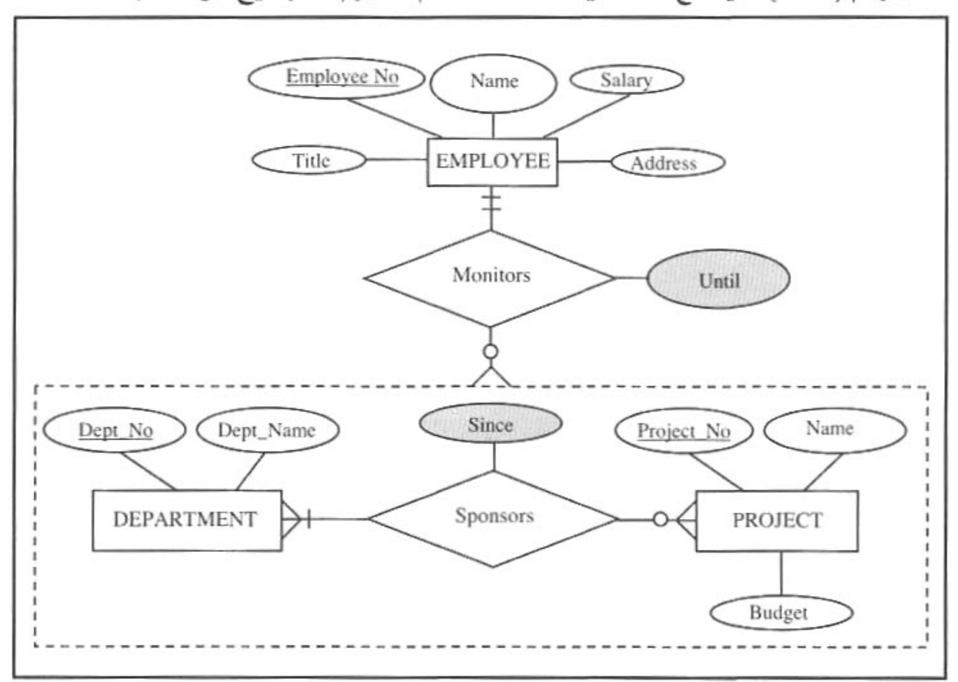
يمكننــا - إذن - التجميع مــن معاملة علاقة ما، مثل «الدعــم المالى» (Sponsors)، وكأنهــا فئة كينونة عند تعريف علاقة أخرى، مثل «المتابعة» (Monitors) أعلاه. ويمكن اســتخدام التجميع، بشكل عام، عند محاولة نمذجة علاقة من ضمن أطرافها علاقة (أو علاقــات) أخــرى. ولكن هل بالإمكان عدم اســتخدام التجميع عندما يكون أحد أطراف علاقة ما علاقة أخرى والاســتعاضة عنه باســتخدام علاقة ثلاثية أو علاقة ثنائية؟ والإجابة عن هذا السؤال هو عدم إمكانية ذلك فى مثل الحالة التى افترضناها أعلاه؛ إذ إنه لا يمكن استخدام علاقة ثلاثية تربط بين الكينونات الثلاث؛ وذلك لوجود علاقتين مستقلتين هما «الدعم المالى» و«المتابعة». ليس هذا فحسب، ولكن كل علاقة

مــن العلاقتين قد يكون لهــا خصائصها المتعلقة بها مثل تاريــخ «بداية» الدعم المالى (Since) للمشــروع التى ترتبــط بعلاقة «الدعم المالى» وخاصيــة «نهاية» فترة المتابعة (Until) للموظف التى ترتبط بعلاقة «المتابعة».

كمــا أنه لا يمكن الاســتعاضة عــن التجميع من خلال علاقــة ثنائية وهى علاقة «المتابعة» التى تربط بين كينونة المشــروع وكينونة الموظف، على ســبيل المثال. ويعزى السبب وراء ذلك لكون علاقة المتابعة لا ترتبط بالمشروع ولكنها ترتبط بعلاقة «الدعم المالى».

ويمكن إيضاح قيود التعددية على علاقة المتابعة كما لو كان التجميع فئة كينونة. فعلى ســبيل المثال، لنفترض أن الموظف الواحد قــد لا يتابع أى دعم مالى أو أنه يقوم بمتابعة أكثر من دعم مالى واحد، وأن كل دعم مالى تتم متابعته من قبل موظف واحد فقط. فى هذه الحالة، يمكن تمثيل قيود التعددية كما هو موضح فى الشكل رقم (٣-٢١).

شكل رقم (٣-٢١): إيضاح التعددية عند استخدام مفهوم التجميع فى نمذجة العلاقات



الفصل الرابع

النموذج العلاقى ولغاته الرسمية

يركز هذا الفصل من الكتاب على شـــرح المفاهيم الأساسية للنموذج العلاقى الذى يعـد أنجح النماذج التمثيلية للبيانات وأكثرها اســتخداماً فــى نظم قواعد البيانات المتوافرة على المســتوى التجارى. ومن أبرز أســباب نجاح وانتشـــار اســتخدام هذا النموذج ســـهولته فى تمثيل البيانات بالإضافة لاســتناده إلى أســس رياضية صلبة تمكنه من التعامل مع البيانات وحســاب نتائجها. لذلك فإن هذا الفصل يســتعرض أيضاً لغتين من لغات النموذج العلاقى الرســمية وهمــا: الجبر العلاقى (Relational Algebra) – التـــى تعد إحدى لغات النموذج العلاقى الإجرائية (Procedural Language)؛ والحساب العلاقى (Relational Calculus) – التى تعد إحدى لغات النموذج العلاقى غير الإجرائية (Nonprocedural Language). كما أن من أسباب نجاح هذا النموذج وجود لغة قياســية (Standard Language) للتعامل مع هذا النموذج، وهى لغة الاستفسار البنائية (Structured Query Language (SQL)) التى يمثل شـــرح مكوناتها محتوى كل من الفصل السابع والفصل الثامن.

٤-١ نموذج البيانات العلاقى (Relational Data Model):

ظهر النموذج العلاقى عام ١٩٧٠م على يد إدغار كود الذى كان يعمل فى شركة آى بى إم (IBM)، وذلك من خلال ورقته العلمية الشـــهيرة التى قام بنشـــرها عام ١٩٧٠م (Codd, 1970). ولقد لاقى هذا النموذج اهتماماً كبيراً منذ بداية ظهوره وذلك لسهولته فى تمثيل البيانات ولوجود أســـس رياضية يعتمد عليهـــا. وتم البدء فى بناء نظامين تجريبيين بحثيين للتأكد من جدوى النموذج العلاقى: الأول منهما كان فى أحد مراكز شركة آى بى إم للبحوث (IBM San Jose Research Laboratory) الذى تم فيه تطوير نظام عرف بنظام «آر» (System R) فى أواخر الســـبعينيات الميلادية، أما الثانى فكان نظاماً ذا طبيعـــة أكاديمية وتم تطويره فى جامعـــة بيركلى الأمريكية وعرف بنظام «إنجرس» (Ingres). ومع بداية الثمانينيات الميلادية بدأت تظهر نظم إدارة قواعد بيانات عديدة تعتمد فى بنائها على النموذج العلاقى. أما اليوم فإن نظم قواعد البيانات العلاقية هى

السائدة، وهى تُراوح فى استخداماتها بين تلك التى يمكن تركيبها واستخدامها على الحاسبات الشخصية وصولاً إلى تلك التى يتم تركيبها واستخدامها على «الحاسبات الكبيرة» (أو «المركزية») (Mainframes).

٤-١-١ المفاهيم الأساسية فى النموذج العلاقى:

تمثل البيانات فى نموذج البيانات العلاقى على هيئة مجموعة من الجداول تسمى علاقات (Relations). وكل جدول فى النموذج العلاقى يحتوى على مجموعة من الصفوف ومجموعة من الأعمدة بحيث إن كل صف من صفوف الجدول يمثل بيانات حالة من الحالات التى لها مفهومها فى المنظمة، وكل عمود يمثل إحدى خصائص هذه الحالة. ويتكون النموذج العلاقى من ثلاثة مكونات رئيسية (Fleming and von Halle, 1989) وهى:

١- **هياكل البيانات (Data Structures):** يتم تنظيم البيانات ضمن جداول (تسمى رسمياً علاقات فى النموذج العلاقى) تتكون من صفوف (تسمى (Tuples) باللغة الإنجليزية) وأعمدة (تسمى (Attributes) باللغة الإنجليزية).

٢- **عمليات للتعامل مع البيانات:** يتوافر للنموذج العلاقى لغات تمكن من التعامل مع بيانات النموذج. وهذه اللغات مبنية على أسس رياضية صلبة (مثل الجبر العلاقى والحساب العلاقى).

٣- **تكامل البيانات:** يتوافر فى النموذج العلاقى تسهيلات تمكن من توصيف (بعض) قواعد العمل التى تمكن من المحافظة على تكامل البيانات.

وفيما يلى شرح لهياكل البيانات وتكامل البيانات فى النموذج العلاقى. أما لغات التعامل (الرسمية) مع النموذج العلاقى فسيتم شرحها فى الجزأين التاليين من هذا الفصل.

٤-١-١-١ هيكل البيانات العلاقى:

تُمثل العلاقة الهيكل الأساسى للبيانات فى النموذج العلاقى الذى يستمد مسماه من مسَمى الهيكل الرئيسى لبياناته. والعلاقة هى جدول بسيط له مسمى. وتتكون العلاقة (أو الجدول) من عدد محدد من الحقول (أو الأعمدة) لها مسميات وعدد غير محدد من الصفوف دون مسميات لها. ويُمثِل عدد الحقول «درجة الجدول»

((Degree (or arity)). وكل صــف فى الجدول يمثل ســجلاً (Record) لإحدى الحالات بحيــث يحتوى على قيــم للبيانات فى كل عامــود من الأعمدة التــى يحتوى عليها الجدول. ويعنى هذا أن كل عمود يمثل إحدى خصائص الحالات المدونة فى الجدول ويحمل اســم الخاصية. ويوضح الشــكل رقم (٤-١) جدول «عضو هيئة التدريس» (FACULTY_T) وفق النموذج العلاقى، الذى يكافئ فئة كينونة «عضو هيئة التدريس» فى النموذج كينونة - علاقة.

شكل رقم (٤-١): جدول «عضو هيئة التدريس» (FACULTY_T) وفق النموذج العلاقى

FACULTY_T

Faculty_ID	FName	LName	Phone_No	Salary	DOB
200	Khalid	Aloufi	454-2341	35000	22/05/1963
220	Fahad	Alhamid	456-7733	25900	07/10/1970
310	Saleh	Aleesa	454-8932	30000	13/09/1966
320	Mohammed	Alhamad	454-5412	44000	13/05/1965
330	Ghanim	Alghanim	456-2234	44500	12/08/1969
340	Ibraheem	Alsaleh	454-1234	25000	20/01/1970
400	Ahmad	Alotaibi	454-4563	33900	17/05/1971
420	Saleh	Alghamdi	454-2233	44600	13/02/1969
500	Yahya	Khorshid	456-2221	36700	12/03/1965
540	Salem	Alhamad	456-3304	40000	11/09/1972
560	Salman	Albassam	454-7865	33800	13/09/1968
600	Turki	Alturki	456-7891	27800	23/07/1975
640	Fahad	Alzaid	456-3322	44300	12/05/1971
660	Saud	Alkhalifa	454-9856	44900	13/08/1972
710	Mahmood	Alsalem	456-3323	31900	19/02/1973
730	Mishal	Almazid	454-2343	29800	17/09/1975
770	Sultan	Aljasir	456-3212	43300	13/05/1970
800	Ali	Albader	456-7812	45300	22/06/1966
810	Saad	Alzhrani	454-5578	44200	17/10/1967
850	Ahmad	Alsabti	456-0120	33900	15/04/1973

ويتكون الجدول الســابق من ستة حقول تمثل الخصائص التالية: رقم عضو هيئة التدريس، والاســم الأول لعضو هيئة التدريس، واســم العائلة، ورقم الهاتف، والمرتب، وتاريــخ الميلاد. وعليه فــإن درجة الجدول هى ســتة (٦). أما كل صف من صفوف

الجـــدول فيمثل حالـــة واحدة من حالات أعضاء هيئة التدريـــس. وقد تم إدراج عدد من حالات أعضـــاء هيئة التدريس فى الجدول لإيضاح هيكل الجدول؛ إذ إن الحالات نفسها لا تعد من ضمن هيكل الجدول. كما أن هذه الحالات تتغير بشكل مستمر من خـــلال عمليات الإضافة، والحذف، والتعديل. ويعنى هذا أن الســـجلات الموجودة فى الجـــدول، فى أية لحظة ما، تمثل حالـــة من حالات الجدول فى تلك اللحظة والتى قد تتغير فى لحظـــة أخرى. ومن الممكن أن تكون حالة الجدول فارغة بمعنى عدم وجود أية سجلات فيه.

ويتم فى بعض الأحيان تمثيل هيكل الجدول بشـــكل مختصر من خلال كتابة اســـم الجدول متبوعاً بأسماء حقول الجدول التى تتم كتابتها بين قوسين. فعلى على سبيل المثال، يتم تمثيل جدول «أعضـــاء هيئة التدريس» أعلاه وفق الطريقة المختصرة هذه، كما يلى:

FACULTY_T (Faculty_ID, FName, LName, Phone_No, Salary, DOB)

٤-١-١-٢ المفاتيح فى النموذج العلاقى:

يُعَرف الجدول (أو العلاقة) فى النموذج العلاقى على أنه مجموعة من الســـجلات (Tuples). ولأن المجموعات فى تعريفها الرياضى لا تحتوى على قيم متكررة، يجب فى الســـجلات التى تخزن فى أى جدول علاقـــى ألا تتكرر فى نفس الجدول حتى ينطبق عليها تعريف المجموعة. ويعنى هذا أنه لا يمكن أن يكون لســـجلين فى جدول علاقى نفـــس التوليفات فى قيم خصائصهمـــا. ويوجد عادة مجموعة من الخصائص فى أى جدول علاقى لا يمكن أن تتكرر بين ســـجلات الجدول. وتسمى هذه الخصائص التى تميز بين السجلات المختلفة فى الجدول «المفتاح الخارق» (Superkey (SK)). ويعنى هذا أنه لأى سجلين (Tuples)، وليكونا t_1 و t_2، فى جدول علاقى ما، وليكن اسمه "R"، وعلى افتراض أن المفتاح الخارق للجدول هو "SK" فإنه لا بد أن يتحقق الشرط التالى:

$$t_1 [SK] \neq t_2 [SK]$$

ويعنى الشـــرط أعلاه أنه لا يمكن أن يوجد فى جـــدول علاقى ما، له مفتاح خارق اسمه "SK" يتكون من مجموعة من الخصائص، أن يتكرر فى سجلين هما t_1 و t_2. كما

يعنى هذا أن أى مجموعة من الخصائص فى الجدول تحقق الشرط أعلاه تعد مفتاحاً خارقاً للجدول. وذلك يعنى أنه بالإمكان أن يتوافر فى الجدول أكثر من مفتاح خارق. ويضمن مبدأ المفاتيح الخارقة تفرد السجلات فى الجداول العلاقية بحيث إنه لا يمكن أن يوجد سجلان فى جدول ما، وفى أية حالة من حالات الجدول، أن يكون لهما نفس قيمة المفتاح الخارق. ولكل جدول علاقى مفتاح خارق واحد على الأقل. وبذلك يكون المفتاح الخارق الافتراضى لأية جدول علاقى مكوناً من كافة حقول الجدول.

ولعـدم نص تعريف المفتاح الخارق على أن الحقول المكونة للمفتاح الخارق يجب أن تكون أقل ما يمكن من حقول تُمَكن من التعرف على سجلات الجدول بشكل منفرد، فإنه قد يكون من ضمن الحقول المكونة للمفتاح الخارق حقول لا تؤثر فى عملية التعرف على سجلات الجدول بشكل منفرد. لذا فإن هذا يقودنا إلى تعريف مبدأ «المفتاح» الذى يعد أكثر فائدة من مبدأ المفتاح الخارق. و«المفتاح» فى أية علاقة هو «مفتاح خارق» ولكننا لا نستطيع حذف أى حقل من الحقول المكونة له، وفى الوقت نفسه، الاستمرار فى التعرف على سجلات الجدول بشكل منفرد. لذا فإن أى مفتاح لجدول علاقى ما يجب أن يحقق الشرطين التاليين:

١- لا يمكن أن تتكرر قيم (كافة) الحقول المكونة للمفتاح، فى أية حالة من حالات الجدول، لسجلين مختلفين فى الجدول.

٢- لا يمكن حذف أى حقل من الحقول المكونة للمفتاح، وفى الوقت نفسه، الاستمرار فى التعرف على سجلات الجدول بشكل منفرد.

ينطبق الشرط الأول على كل من «المفتاح الخارق» و «المفتاح». أما الشرط الثانى فينطبق على «المفتاح» فقط. كما يعنى الشرطان مع بعضهما أن «المفتاح» هو «مفتاح خارق»، يميز بين سجلات الجدول بشكل منفرد، ولكنه يتكون من أقل عدد ممكن من الحقول التى تميز بين سجلات الجدول بشكل منفرد وفى أية حالة من حالات الجدول. فعلى سبيل المثال، يمثل حقل «رقم عضو هيئة التدريس» (Faculty_ID) مفتاحاً لجدول «أعضاء هيئة التدريس» أعلاه لكونه يميز بين سجلات الجدول، المتعلقة بأعضاء هيئة التدريس، بشكل منفرد ليس فى حالة الجدول أعلاه فحسب ولكن فى أية حالة ممكنة من الحالات التى قد يكون عليها الجدول. كما يعد رقم عضو هيئة التدريس مع أية حقول أخرى فى الجدول مفتاحاً خارقاً. فعلى سبيل المثال، يعد حقل رقم عضو هيئة التدريس والاسم الأول مفتاحاً خارقاً للجدول، ورقم عضو هيئة التدريس واسم العائلة

مفتاحاً خارقاً آخر للجدول، ورقم عضو هيئة التدريس والاسم الأول والمرتب مفتاحاً خارقاً ثالثاً للجدول، ... وهكذا. وهذا يعنى أن كل «مفتاح» هو «مفتاح خارق» ولكن ليس كل «مفتاح خارق» يكون «مفتاحاً». فعلى سبيل المثال، عند حذف حقل الاسم الأول من المفتاح الخارق المكون من حقل رقم عضو هيئة التدريس والاسم الأول، ما زلنا نستطيع التعرف على كل سجل من سجلات الجدول بشكل منفرد. لذا فإن الشرط الثانى أعلاه لا ينطبق على هذا المفتاح وبالتالى فإنه «مفتاح خارق» ولكنه ليس «مفتاحاً» للجدول. تجدر الإشارة هنا أنه ليس من الضرورى أن يتكون أى مفتاح (سواء كان خارقاً أو مفتاحاً فقط) من حقل واحد فقط؛ إذ إنه فى الكثير من الأحيان يكون مفتاح الجدول «مفتاحاً مركباً» يتكون من أكثر من حقل من حقول الجدول.

ويعد وجود مفتاح فى أى جدول علاقى خاصية من خصائص هياكل الجداول فى النموذج العلاقى بدونها لا يعد الجدول علاقياً. ويمكن تحديد المفتاح لأى جدول من خلال خصائص الحقول المكونة له بحيث يجب أن تكون من ضمن خصائص هذه الحقول عدم تغير قيمها بتغير الزمن بالإضافة لكونها تميز بين سجلات الجدول بشكل منفرد. فعلى سبيل المثال، لا يمكن أن يكون رقم الهاتف مفتاحاً لجدول أعضاء هيئة التدريس؛ لأنه قد يتغير بتغير الزمن. كذلك هو الحال بالنسبة لحقل الاسم الأول واسم العائلة، فهذان الحقلان مدمجان مع بعضهما لا يصلحان أن يصبحا مفتاحاً للجدول؛ لأنهما قد يتكرران فى أكثر من سجل من سجلات أعضاء هيئة التدريس. أما رقم عضو هيئة التدريس فله قيمة لا تتكرر بين أعضاء هيئة التدريس، وفى الوقت نفسه، لا تتغير بتغير (أو مرور) الزمن.

وبشكل عام، فإنه قد يوجد فى هيكل أى جدول علاقى أكثر من مفتاح. فى هذه الحالة يعد كل مفتاح من هذه المفاتيح «مفتاحاً مرشحاً». فعلى سبيل المثال، وفى حالة تم إدراج رقم السجل المدنى ليصبح حقلاً ضمن جدول أعضاء هيئة التدريس أعلاه، فإن كلاً من رقم عضو هيئة التدريس ورقم السجل المدنى لعضو هيئة التدريس يعدان مفاتيح مرشحة فى جدول أعضاء هيئة التدريس، تنطبق على كل منهما شروط المفتاح. وإذا وُجِدَ أكثر من مفتاح لجدول ما، فإنه يتم اختيار أحد هذه المفاتيح المرشحة ليصبح «المفتاح الرئيسى» للجدول. ويعنى هذا أن «المفتاح الرئيسى» هو أحد المفاتيح المرشحة الذى تم اختياره ليميز بين السجلات المختلفة فى الجدول. ومن المتعارف عليه عند اختيار المفتاح الرئيسى من ضمن مجموعة من المفاتيح المرشحة، هو اختيار المفتاح المرشح ذى العدد الأقل من الحقول. وعند تمثيل المفتاح الرئيسى فى هيكل الجدول،

جـرت العـادة على أن يوضع خط تحت الحقـل المكون (أو الحقـول المكونة) للمفتاح الرئيسى، وذلك للتمييز بين حقول المفتاح الرئيسى وبقية حقول الجدول. أما بالنسبة لبقية المفاتيح المرشحة، فإنها تسـمى أيضاً «مفاتيح ثانوية» (Secondary Keys) قد يتم استخدام بعض منها فى إنشاء فهارس ثانوية للجدول أثناء عملية بناء قاعدة البيانات (أثناء مرحلة التصميم المادى)؛ وذلك بهدف التسـريع فى عملية البحث فى سـجلات الجدول واسترجاعها.

٤-١-١-٣ خصائص العلاقات (أو الجداول) فى النموذج العلاقى:

تم تعريـف العلاقة أعلاه على أنها جدول بسـيط، ولكنه من الضرورى معرفة أنه ليـس كل جدول بسـيط يمثل علاقة وفقاً للنمـوذج العلاقى. ولهذا السـبب لم يتم اسـتخدام كلمة «جدول» فى النموذج العلاقى بشكله الرسمى، وإنما تم استخدام كلمة «علاقة». فالعلاقات لها عدد من الخصائص التى تميزها عن الجداول غير العلاقية (أو التقليدية) التى نتعامل معها فى حياتنا اليومية. وفيما يلى اسـتعراض لخصائص العلاقات (أو الجداول العلاقية):

١- كل جدول علاقى يجب أن يكون له اسم.
٢- كل خاصية (أو عمود) لها اسـم فريد يميزها عـن بقية الخصائص (أو الأعمدة) المعرفة فى نفس الجدول.
٣- كل صف فى الجدول يعد فريداً لا يمكن أن يتكرر ضمن الجدول.
٤- كل تقاطـع بـين صف وعمود (خلية من خلايا الجـدول) يحتوى على قيمة واحدة فقط، بمعنى أنه لا يمكن إدخال أكثر من قيمة داخل نفس الخلية.
٥- لا يعـد ترتيب الحقول مهمـاً؛ إذ إنه يمكن تغيير ترتيبهـا دون الإخلال بمعنى أو طريقة استخدام الجدول.
٦- لا يعد ترتيب السـجلات (أو صفوف) الجدول مهماً؛ إذ إنه يمكن تغيير ترتيبها أو تخزينها، كما هو الحال بالنسـبة للحقول، دون الإخلال بمعنى أو طريقة استخدام الجدول.

وتعنى الخصائص السابقة للجداول العلاقية أن كل جدول يحتوى على بيانات تمثل مجموعة من الحقائق. فعلى سبيل المثال، يحتوى جدول «أعضاء هيئة التدريس» على بيانـات تمثل حقائق عن أعضـاء هيئة التدريس. فكل صف فى الجدول يمثل حقيقة

عــن بيانــات أحد أعضاء التدريس. ويعنى هذا أن تغيير ترتيب الصف (أو الســجل) الممثل لأحد أعضاء هيئة التدريس ضمن صفوف الجدول لن يغير من البيانات الخاصة بعضو هيئة التدريس. كذلك هو الحال لو تم تغيير ترتيب الأعمدة (أو الحقول) المكونة للجدول، فإن مثل هذا التغيير لن يغير من البيانات التى تمثل حقائق عن أعضاء هيئة التدريس. أما بالنســبة لمســميات الأعمدة (أو الحقول) فى جداول النموذج العلاقى فهــى مهمة؛ لأنها تدل على معنى البيانات (أو الحقائق) التى يحتويها كل حقل وكذلك للتفريــق بين معنى حقل مــا وبقية الحقول فى الجدول. ووجود مســمى لكل جدول يمكننــا من التعامل مع الحقائــق المختلفة الموجودة فى قاعدة البيانات من خلال لغات تداول البيانات فى النموذج العلاقى التى ســنتطرق لها لاحقاً. وبدون هذه المسميات للجداول لا يمكن التعرف على الجدول الذى سيتم التعامل معه.

وتجــدر الملاحظة أن الجداول العلاقية قد تمثل حقائق عن الكينونات أو العلاقات بين الكينونات حسب تعريفها فى نموذج كينونة - علاقة. فعلى سبيل المثال، يتم تمثيل فئة كينونة «عضو هيئة التدريس» وكينونة «القســم الدراسي» كجدولين منفصلين فى النموذج العلاقى لتمثيل الحقائق التى تتعلق بكل من أعضاء هيئة التدريس والأقســام الدراســية، علــى التوالى. أما العلاقة التى تربط بــين الكينونتين وهى علاقة «يعمل فــى» فإنه يتم تمثيلها أيضاً لتصبح جدولاً فى النموذج العلاقى تمثل الحقائق المتعلقة بالقســم الدراســى الذى يعمل فيه كل عضــو هيئة تدريس. ويعنى هــذا أن كلاً من الكينونات والعلاقات (ذات تعددية متعدد - متعدد) تُمَثل فى النموذج العلاقى بشــكل متماثل على هيئة جداول.

٤-١-١-٤ قيود التكامل (Integrity Constraints) فى النموذج العلاقى:

تحتوى أية قاعــدة بيانات علاقية على العديد من الجداول. وترتبط الســجلات الموجودة فى الجداول المختلفة المكونة لقاعدة البيانات بأشــكال متنوعة حتى تستطيع أن تمثــل الواقــع الذى نحاول تمثيلــه. وتُمثِل كل «حالة» من حــالات قاعدة البيانات «حالات» كافة جداول قاعدة البيانات فى لحظة معينة من الزمن. ويوجد عادة الكثير مــن القيود التى يجب أن تنطبق على قاعدة البيانات فى أية حالة من حالاتها. وتُمثِل هــذه القيود قواعــد العمل المعمول بها فــى المنظمة التى نحاول تمثيــل بياناتها وفق النموذج العلاقى. فعلى ســبيل المثال، عندما نقول أن كل عضو هيئة تدريس لا بد أن يتبع (أو يعمل) فى قسم دراسى واحد فقط، فإن مثل قاعدة العمل هذه تمثل قيداً على

كافة حالات قاعدة البيانات. ويعنى هذا أنه فى أية حالة من حالات قاعدة البيانات لا بد أن يرتبط كل عضو هيئة تدريس بقسم دراسى يمثل مقر عمله حتى تكون قاعدة البيانات صحيحة ومتكاملة. وفيما يلى شرح لأهم أنواع القيود التى يمكن تمثيلها فى النموذج العلاقى. وهذه القيود هى: قيود المجال (Domain Constraints)، وقيود تكامل الجدول (Entity Integrity Constraints)، وقيود على القيم غير المعرفة (NULL Constraints)، وقيود السلامة المرجعية (Referential Integrity Constraints).

٤-١-١-٤-١ قيود المجال (Domain Constraints):

عندما يتم تعريف حقل ضمن جدول علاقى فإنه لا بد أن يتم تحديد نوعية البيانات التى سيتخذ الحقل قيماً منها. وهذه الحالة شبيهة بتعريف المتغيرات (Variables) فى لغات البرمجة؛ إذ إن أى متغير لا بد أن يتم تحديد نوعية لبياناته (وذلك فى الغالبية العظمى من لغات البرمجة). أما المجال فهو مجموعة محددة (أو مدى) من القيم التى من الممكن أن يتخذ الحقل قيماً منها. لذا فإن أى مجال فى النموذج العلاقى لا بد أن يرتبط بنوعية بيانات محددة وبمدى محدد. فعلى سبيل المثال، يمكن تحديد نوعية بيانات حقل «راتب عضو هيئة التدريس» فى جدول أعضاء هيئة التدريس على أنه من نوع «الأعداد الصحيحة» (Integers). إلا أنه فى غالبية المنظمات يوجد حد أعلى وحد أدنى للرواتب التى يتقاضاها الموظفون. فى هذه الحالة يمكن تحديد مدى القيم المسموح بها لأن تكون رواتب لأعضاء هيئة التدريس وليكن بين ٥,٠٠٠ و ٣٠,٠٠٠. وبهذه الطريقة فإن النموذج العلاقى يمكن من وضع قيود على مدى القيم التى من الممكن أن يأخذها أى حقل، مما يسهم فى صحة البيانات وتكاملها؛ لأن محاولة إدخال أية قيمة تخرج عن المجال، أى نوعية البيانات ومدى القيم المسموح بها، الذى تم تحديده لحقلٍ ما لن تكون مقبولة فى النموذج العلاقى.

ويتكون تعريف أى مجال لحقل ما، فى العادة، من: اسم المجال (Domain Name)، ومعناه (Meaning)، ونوعية بياناته (Data Type)، وطوله (Size)، ومجموعة (أو مدى) القيم المسموح بها. ويسهم استخدام اسم لأى مجال فى تسهيل عملية تفسير القيم التى من الممكن أن يأخذها أى حقل يرتبط به. ويوضح الجدول رقم (٤-١) تعاريف لمجال بعض الحقول.

جدول رقم (٤-١): أمثلة لتعاريف مجال بعض الحقول

الحقل (أو العمود)	«اسم المجال» الذى يرتبط به الحقل	معنى المجال	مدى القيم
Department_ID	Department_IDs	مجموعة القيم التى من الممكن أن يأخذها حقل «رمز القسم الدراسى»	نصى بطول ستة أحرف: Char(6)
Gender	Gender_Type	جنس عضو هيئة التدريس: ذكراً أو أنثى	نصى بطول حرف واحد من القيم "M" للذكر و "F" للأنثى: Char(1) In ('M', 'F')
Salary	Salary_Range	راتب عضو هيئة التدريس	عدد صحيح بين ٥,٠٠٠ و٣٠,٠٠٠: Integer Between 5,000 and 30,000
Course_ID	Course_IDs	رمز المادة الدراسية	نصى بطول سبعة أحرف: Char(7)

٤-١-١-٤-٢ قيود تكامل الجدول (أو العلاقة) (Entity Integrity Constraints):

إن قيد تكامل الجدول (أو العلاقة) قد تم تصميمه لتأكيد أن كل جدول فى النموذج العلاقى يجب أن يحتوى على مفتاح رئيسى (يتكون من حقل واحد أو أكثر) وبحيث يُمكن هذا المفتاح من التمييز بين الحالات التى يحتويها الجدول (أى السجلات أو الصفوف) فى أية حالة من الحالات التى من الممكن أن يكون عليها. كما أن قيمة المفتاح الرئيسى لأى سجل يجب أن تكون صحيحة. وبشكل خاص، يجب أن لا تكون قيمة أى حقل من الحقول المكونة للمفتاح الرئيسى قيمة غير معرفة (NULL Value). وسبب ذلك يرجع إلى أن المفتاح الرئيسى يجب أن يمكن من التعرف على السجلات المختلفة فى الجدول بشكل منفرد (دون تكرار) ووجود قيمة غير معرفة ضمن أحد حقول المفتاح الرئيسى لن يمكننا من التمييز بين سجلات الجدول بشكل منفرد. فعلى سبيل المثال، لو كان المفتاح الرئيسى (أو أحد حقوله) ذا قيمة غير معرفة فى أكثر من سجل واحد من سجلات جدول ما، فإننا لن نستطيع التمييز بين هذه السجلات لو حاولنا الرجوع إليها من جدول آخر.

٤-١-١-٤-٣ قيود القيم غير المعرفة (NULL Constraints):

فــى بعــض الأحيان قد لا يمكن وضــع قيمة محددة فى حقل مــا. وقد يكون من أسباب ذلك كون الحقل لا ينطبق على الحالة التى نرغب فى إدراجها ضمن جدول ما. فعلى ســبيل المثال، قد يترك حقل «رقم الهاتف» فى سجل أحد أعضاء هيئة التدريس فارغاً (بقيمة غير معرفة) لكون عضو هيئة التدريس الذى نحاول إدراج سجل له ضمن سجلات أعضاء هيئة التدريس ليس لديه خط هاتفى. فى مثل هذه الحالة لا ينطبق هــذا الحقل على عضو هيئة التدريس، ولذلــك يترك فارغاً للدلالة على هذا الوضع. ومن الأســباب الأخرى التى تستدعى استخدام قيم غير معرفة فى حقل ما هو عندما تكون قيمة الحقل موجودة على أرض الواقع ولكنها غير متوافرة وقت إدخال السجل. فعلى سبيل المثال، قد يتم إدخال سجل لعضو هيئة التدريس دون معرفة تاريخ ميلاده ليــس لأن عضو هيئــة التدريس ليس لديه تاريخ ميلاد ولكــن لكون تاريخ الميلاد غير متوافر وقت إدخال ســجل عضو هيئة التدريــس. فى مثل هذه الحالة يتم ترك حقل تاريخ الميلاد فارغاً (بقيمة غير معرفة) للدلالة على هذا الوضع.

ويمكن النموذج العلاقى من إدخال قيم غير معرفة (NULL Values) فى مثل الحقول التى تم الإشارة إليها سابقاً سواء بشكل ضمنى من خلال تركها فارغة (أى عدم إدخال قيمة فيها) أو من خلال إدخال القيمة "NULL" بشكل صريح. وفى الواقع فإن القيمة غيــر المعرفة (NULL) ليســت قيمة ولكنهــا للدلالة على «غياب قيمــة محددة» يمكن إدخالها فى الحقل. لذلك فإن القيمة غير المعرفة ليســت مثل العدد صفر أو سلسلة حرفية فارغة (Blank String). وعلى الرغم من أهمية مبدأ القيم غير المعرفة فى تمثيل البيانات فى النموذج العلاقى إلا أنها تعد مصدراً من مصادر الالتباس، كما ســنوضح فى الفصل الســابع (المتعلق بلغة الاستفسار البنائية)، وذلك لكون القيمة غير المعرفة هى واقعياً ليســت قيمة، كما أســلفنا أعلاه، ولكنها للدلالة على غياب القيمة. ويعنى هــذا أنه لا يمكن مقارنة قيمة غير معرفة فى حقل ما مع قيمة غير معرفة أخرى فى نفس الحقل ولكن لسجل آخر.

ويفيــد قيد القيم غير المعرفة فــى تقييد القيم التى من الممكن أن يأخذها الحقل، بحيــث لا تكــون القيمة «غير المعرفــة» (NULL) من ضمن القيم التــى من الممكن أن يأخذهــا الحقل. فعلى ســبيل المثال، لا يمكن أن يُدخل ســجل لأحــد أعضاء هيئة التدريس دون أن يتوافر له اســمٌ أولاً واســمٌ لعائلته. فى مثل هذه الحالة يمكن تقييد مثل هذين الحقلين (وهما الاســم الأول واســم العائلة) بحيث لا يمكن أن تكون أى من قيمهما قيمة غير معرفة من خلال وضع القيد "NOT NULL" على كل منهما.

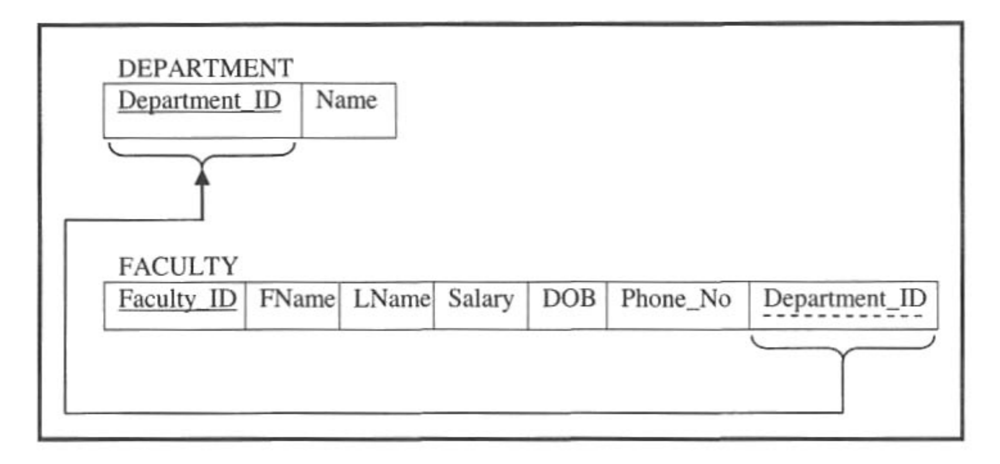
٤-١-٤-٤ قيود السلامة المرجعية (Referential Integrity Constraints):

إن قيد وجود مفتاح رئيسى لكل جدول فى النموذج العلاقى وقيد تكامل الجدول (من حيث وجوب أن تكون قيم المفتاح الرئيسى معرفة دائماً) تنطبق على كل جدول على حدة. على النقيض من ذلك، فإن قيود السلامة المرجعية تتعلق بعملية الربط بين جدولين. وتضمن قيود السلامة المرجعية المحافظة على تناسق البيانات بين السجلات التابعة للجدولين. لذلك فإن قيد السلامة المرجعية ينص على أن أى سجل فى أحد الجدولين يشير إلى سجل فى الجدول الآخر، فإنه يجب أن يشير إلى سجل موجود فعلاً فى الجدول الآخر أو أن تكون قيمته غير معرفة.

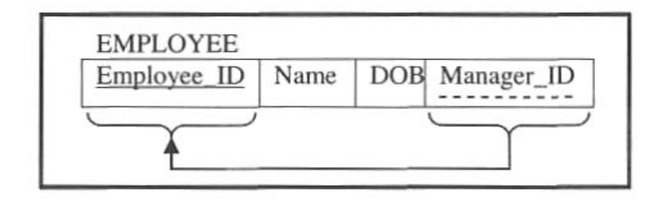
وتمثل عملية الربط بين سجلات الجدولين بما يعرف «المفتاح الخارجى» (Foreign Key). وعند تعريف مفتاح خارجى فى جدول ما، فإن الحقل (أو الحقول) المكونة للمفتاح الخارجى لا تعد جزءاً من خصائص (أو بيانات) الحالة التى يمثلها السجل نفسه ولكنها جزء من خصائص (أو بيانات) حالة أخرى سواء فى نفس الجدول أو فى جدول آخر. وتكون قيمة المفتاح الخارجى فى السجل هى قيمة لأحد المفاتيح الرئيسية لسجل آخر سواء كان فى نفس الجدول أو فى جدول آخر. فعلى سبيل المثال، تمثل العلاقة «يعمل فى» (Works_for) التى تربط بين كل عضو هيئة تدريس والقسم الذى يتبعه (أو يعمل فيه) عضو هيئة التدريس من خلال إدراج حقل جديد داخل جدول أعضاء هيئة التدريس بحيث يمثل الحقل الجديد مفتاحاً خارجياً يشير إلى سجل القسم الذى يتبعه عضو هيئة التدريس. ولكون المفتاح الرئيسى فى أى جدول هو المميز بين السجلات المختلفة فى الجدول، فإن الحقل الذى تمت إضافته ليصبح مفتاحاً خارجياً هو المفتاح الرئيسى لجدول الأقسام الدراسية، كما يوضح الشكل رقم (٤-٢).

ويقصد بالخط المتقطع تحت «رمز القسم» (Department_ID) فى جدول أعضاء هيئة التدريس أن هذا الحقل عبارة عن مفتاح خارجى يشير إلى حقل «رمز القسم» فى جدول الأقسام الدراسية، كما هو موضح بالسهم الذى يصل بين الحقلين. وباستخدام مبدأ المفاتيح الخارجية يمكن الربط بين الجداول المختلفة فى قواعد البيانات العلاقية. ويوضح الشكل رقم (٤-٣) عملية ربط سجلات الجدول مع بعضها لتمثيل العلاقة «يدير» (Manages) التى تنص على أن الموظف الواحد يرأسه مدير واحد فقط، فى حين أن الموظف الواحد قد يَرأس صفراً أو أكثر من الموظفين. ولتمثيل هذه العلاقة يتم إضافة حقل جديد ضمن جدول الموظفين بحيث يحتوى هذا الحقل على قيمة تمثل المفتاح الرئيسى لمدير الموظف.

شكل رقم (٤-٢): تمثيل المفاتيح الخارجية فى النموذج العلاقى



شكل رقم (٤-٣): استخدام المفتاح الخارجى لربط سجلات الجدول نفسه ببعضها



وعند تمثيل العلاقات (حسب تعريفها فى نموذج كينونة - علاقة) من خلال المفاتيح الخارجية فإن قيود السلامة المرجعية تنص على ما يلى:

١- يجب أن تُعَرف حقول المفتاح الخارجى بحيث تكون من نفس مجال المفتاح الرئيسى للجدول الذى يشير إليه المفتاح الخارجى.

٢- قيمة المفتاح الخارجى فى أى سجل فى الجدول يجب أن تشير إلى قيمة موجودة فعلاً فى الجدول الذى يشير إلية المفتاح الخارجى أو أن تكون قيمة المفتاح الخارجى غير معرفة (NULL).

ففى المثال الأول أعلاه، يجب أن يكون حقل المفتاح الخارجى، وهو «رمز القسم الدراسى» فى جدول أعضاء هيئة التدريس، من نفس مجال قيم المفتاح الرئيسى، وهو

أيضاً «رمز القسم الدراسى»، فى جدول أعضاء هيئة التدريس. كما يجب أن تكون كل قيمة مدونة فى حقل المفتاح الخارجى لكافة السجلات فى جدول أعضاء هيئة التدريس لها ما يقابلها من مفاتيح رئيسية فى جدول الأقسام العلمية أو أن تكون قيمها غير معرفة. أما فى المثال الثانى، فإنه يجب أن يكون حقل المفتاح الخارجى، وهو «رمز المدير»، من نفس مجال قيم المفتاح الرئيسى، وهو «رمز الموظف»، فى الجدول نفسه. كما يجب أن تكون كل قيمة مدونة فى حقل المفتاح الخارجى لكافة السجلات فى جدول الموظفين لها ما يقابلها من مفاتيح رئيسية فى نفس الجدول أو أن تكون قيمها غير معرفة. ويلاحظ من المثال الثانى أنه ليس من الضرورى أن تكون مسميات حقول المفاتيح الخارجية متطابقة مع مسميات المفاتيح الرئيسية مادامت من نفس المجال، وأن القيم التى تأخذها إما أن تكون موجودة ضمن قيم المفاتيح الرئيسية أو أن تكون غير معرفة، حسب شرطى قيود السلامة المرجعية أعلاه.

والسؤال الذى قد يُطرح هو: متى يمكن أن يُسمح بأن تكون قيم المفتاح الخارجى غير معرفة ومتى لا يسمح بذلك؟ إن الإجابة عن هذا التساؤل تعود بنا إلى مفهوم التعددية فى العلاقات (Cardinality Constraints) التى سبق أن تم شرحها فى الفصل الثالث. فعندما تكون العلاقة إجبارية (سواء إجبارى واحد أو إجبارى متعدد)، بمعنى أن القيمة الدنيا للتعددية هى واحد فإن المفتاح الخارجى لا يمكن أن يأخذ قيمة غير معرفة. ففى المثال الأول أعلاه، لا يمكن أن تكون قيمة حقل المفتاح الخارجى غير معرفة لأن كل عضو هيئة تدريس يجب أن يعمل فى قسم دراسى. أما فى المثال الثانى، وعلى افتراض أن رئيس المنظمة لا يرأسه أحد، فإن كل موظف يجب أن يرتبط بمدير له تكون قيمة مفتاحه الرئيسى ضمن حقل المفتاح الخارجى للموظف ما عدا رئيس المنظمة الذى ستكون قيمة حقل المفتاح الخارجى فى سجله غير معرفة. ففى مثل هذه الحالة، يمكن أن تكون قيمة حقل المفتاح الخارجى غير معرفة. وبناء على إمكانية أن يأخذ أى مفتاح خارجى القيمة غير المعرفة من عدم إمكانية ذلك، فإنه يمكن توصيف ذلك أثناء عملية إنشاء قاعدة البيانات.

٤-١-١-٤-٥ التعامل مع اختراق القيود أثناء عمليات التعديل على قاعدة البيانات:

يركز هذا الجزء على عمليات التعديل على قاعدة البيانات وعلى إمكانية اختراق هذه العمليات للقيود المفروضة على قاعدة البيانات، وعلى الطرق التى تتعامل معها نظم إدارة قواعد البيانات للمحافظة على سلامة البيانات وتكاملها. ويوفر النموذج

العلاقى ثلاث عمليات تعديل هى: عملية الإضافة (Insert Operation)، وعملية الحذف (Delete Operation)، وعملية التحديث (Update Operation). وتستخدم عملية الإضافة لإضافة سجل أو مجموعة من السجلات لجدول ما، فى حين تستخدم عملية الحذف لحذف سجل أو أكثر من سجلات الجدول. أما عملية التحديث فتستخدم لتغيير قيم بعض الحقول فى سجلات الجدول. وعندما تستخدم هذه العمليات فإنه من الضرورى التأكد من عدم اختراق أى من القيود المفروضة على هيكل قاعدة البيانات العلاقية. وفيما يلى شرح للقيود التى من الممكن أن يتم اختراقها عند إجراء كل من عمليات التعديل الثلاث والطرق التى من الممكن أن تتخذ عن محاولة اختراق أى من القيود المفروضة على هيكل قاعدة البيانات.

٤-١-١-٤-٥-١ عملية الإضافة (The Insert Operation):

تزود عملية الإضافة مجموعة من القيم لقائمة من حقول سجل جديد بغية إضافته إلى جدولٍ ما. ومن الممكن أن تَخترق عملية الإضافة كلاً من قيود التكامل الأربعة التى تم شرحها أعلاه. فمن الممكن أن تكون القيمة المعطاة لأى من حقول السجل الذى ستتم إضافته غير متوافقة مع مجال الحقل. كذلك هو الحال بالنسبة لقيد تكامل الجدول، إذ إن السجل الجديد قد يحتوى على مفتاح رئيسى يتكرر مع سجل موجود أصلاً ضمن الجدول أو أن المفتاح الرئيسى للسجل الجديد ذو قيمة غير معرفة. كذلك هو الحال بالنسبة لقيد القيم غير المعرفة، فقيمة أحد الحقول قد تكون غير معرفة على الرغم من أن القيد المفروض على الحقل الذى سيأخذ هذه القيمة لا يسمح بأن تكون القيمة غير معرفة. أما فيما يتعلق بقيد السلامة المرجعية، فإنه قد يتم اختراقه إذا احتوى السجل الجديد على قيمة للمفتاح الخارجى ليس لها ما يقابلها من مفتاح رئيسى فى الجدول الذى من المفترض أن يشير إليه حقل المفتاح الخارجى، أو أن تكون قيمة المفتاح الخارجى المدخلة غير معرفة (NULL) على الرغم من أن المفتاح الخارجى قد تم توصيفه على أساس أنه لا يمكن أن يكون غير معرف. وفيما يلى بعض الأمثلة لمثل هذه الاختراقات وردود فعل نظام إدارة قاعدة البيانات إزاءها:

١- إضافة سجل عضو هيئة تدريس جديد يحتوى على قيمة غير معرفة فى حقل المفتاح الرئيسى: سيقوم النظام برفض عملية الإضافة هذه لاختراقها لقيد تكامل الجدول الذى ينص على أن قيمة المفتاح الرئيسى لأى سجل لا يمكن أن تكون غير معرفة.

٢- إضافة سجل عضو هيئة تدريس يحتوى على قيمة فى حقل المفتاح الخارجى الذى يربطه بالقسم الدراسى الذى يعمل فيه وأن هذه القيمة لا توجد أصلاً ضمن المفاتيح الرئيسية لجدول الأقسام العلمية؛ سيقوم النظام برفض عملية الإضافة هذه لاختراقها لقيد السلامة المرجعية.

٣- إضافة سجل عضو هيئة تدريس يحتوى على قيمة غير معرفة (NULL) فى حقل المفتاح الخارجى الذى يربطه بالقسم الدراسى الذى يعمل فيه؛ سيقوم النظام برفض عملية الإضافة هذه لاختراقها لقيد القيم غير المعرفة، حيث إن كل عضو هيئة تدريس يجب أن يرتبط بقسم يعمل فيه.

٤- إضافة سجل عضو هيئة تدريس يحتوى على قيمة مكونة من سلسلة حرفية ذات عشرة حروف فى حقل رقم الهاتف؛ سيقوم النظام برفض عملية الإضافة هذه لاختراقها لقيد المجال إذا كان حقل رقم الهاتف معرف على أساس كونه سلسلة حرفية بطول ثمانية حروف.

٤-١-١-٤-٥-٢ عملية الحذف (The Delete Operation):

من الممكن أن تخترق عملية الحذف قيد التكامل المرجعى فقط، وذلك إذا كان السجل المزمع حذفه مشاراً إليه من قبل سجلات أخرى ضمن حقول مفاتيحها الخارجية. فعلى سبيل المثال، سيتم اختراق قيد السلامة المرجعية عند محاولة حذف أى سجل من سجلات جدول الأقسام الدراسية، وذلك عندما يكون هناك أعضاء هيئة تدريس يعملون فى القسم الدراسى المزمع حذف سجله؛ لأن قيمة المفاتيح الخارجية فى سجلات أعضاء هيئة التدريس ستكون مساوية للمفتاح الرئيسى للقسم الذى سيتم حذف سجله. ولو افترضنا إمكانية حذف مثل هذا السجل، فإن سجلات أعضاء هيئة التدريس التابعين للقسم الدراسى ستحتوى على قيم فى حقول مفاتيحها الخارجية غير موجودة ضمن المفاتيح الرئيسية لجدول الأقسام العلمية مما يعد اختراقاً لقيد السلامة المرجعية.

يوفر النموذج العلاقى أربعة خيارات عند حدوث خروقات لقيد السلامة المرجعية حيث يتم استخدام الخيار المناسب عند توصيف حقل المفتاح الخارجى أثناء إنشاء الجدول. وسيتم شرح طريقة توصيف هذه الخيارات باستخدام لغة الاستفسار البنائية فى الفصل السابع. أما هذه الخيارات الاربعة فهى كما يلى:

١- رفض عملية الحذف، وهذه الطريقة تسمى عملية «الرفض» (Reject) وهى الحالة الافتراضية عند اختراق قيد التكامل المرجعى نتيجة لعملية حذف. فعلى سبيل المثال، لو تم تعريف قيد التكامل المرجعى لحقل المفتاح الخارجى فى جدول أعضاء هيئة التدريس الذى يشير إلى جدول الأقسام العلمية على أنه من نوع الرفض عند اختراق قيد التكامل المرجعى، فإن حذف أى سجل من سجلات جدول الأقسام العلمية سيترتب عليه رفض عملية الحذف إذا وجد أى سجل من سجلات أعضاء هيئة التدريس يشير إلى سجل جدول الأقسام العلمية المزمع حذفه.

٢- قبول عملية الحذف مع حذف كافة السجلات التى تشير للسجل الذى تم حذفه، وتسمى هذه الطريقة «الحذف المتسلسل» (Cascade). فعلى سبيل المثال، لو تم تعريف قيد التكامل المرجعى لحقل المفتاح الخارجى فى جدول أعضاء هيئة التدريس على أنه من نوع الحذف المتسلسل، فإن حذف أى سجل من سجلات جدول الأقسام العلمية سيترتب عليه حذف كافة سجلات أعضاء هيئة التدريس التابعين لهذا القسم الدراسى.

٣- قبول عملية الحذف مع وضع قيمة غير معرفة فى الحقول التى تشير للسجل المحذوف، وتسمى هذه الطريقة «وضع قيمة غير معرفة» (Set to NULL). فعلى سبيل المثال، لو تم تعريف قيد التكامل المرجعى لحقل المفتاح الخارجى فى جدول أعضاء هيئة التدريس على أنه من نوع وضع قيمة غير معرفة، فإن حذف أى سجل من سجلات جدول الأقسام العلمية سيترتب عليه وضع قيمة غير معرفة فى حقل المفتاح الخارجى لسجلات أعضاء هيئة التدريس الذين يتبعون للقسم الدراسى الذى تم حذف سجله. وتجدر الملاحظة إلى أنه فى مثل هذه الحالة لا يمكن توصيف حقل المفتاح الخارجى فى جدول أعضاء هيئة التدريس على أنه لا يمكن أن يكون «غير معرفٍ» (Not NULL)؛ لأن خيار الحذف هذا سيترتب عليه خرق قيد القيم غير المعرفة.

٤- قبول عملية الحذف مع وضع «قيمة افتراضية» (Default Value) فى الحقول التى تشير للسجل المحذوف، وتسمى هذه الطريقة وضع قيمة افتراضية. فعلى سبيل المثال، لو تم تعريف قيد التكامل المرجعى لحقل المفتاح الخارجى فى جدول أعضاء هيئة التدريس على أنه من نوع وضع قيمة افتراضية، فإن حذف أى سجل من سجلات الأقسام العلمية سيترتب عليه، وفق هذا الخيار، وضع القيمة الافتراضية المصاحبة لحقل المفتاح الخارجى فى كل سجل من سجلات أعضاء هيئة التدريس التابعين للقسم العلمى المزمع حذف سجله.

ويمكن النموذج العلاقى من استخدام أى توليفات من الخيارات الأربعة السابقة. فعلى سبيل المثال، يمكن استخدام الخيار الثالث عند توصيف حقل المفتاح الخارجى فى جدول أعضاء هيئة التدريس الذى يشير لجدول الأقسام العلمية، فى حين يمكن استخدام الخيار الثانى عند توصيف حقل المفتاح الخارجى فى جدول المواد الدراسية الذى يربط كل مادة دراسية بالقسم الدراسى المسئول عن تنفيذ المادة الدراسية. وعند حذف أى سجل من جدول الأقسام الدراسية سيتم وضع قيمة غير معرفة فى حقل المفتاح الخارجى الذى يشير إلى جدول الأقسام الدراسية لكافة سجلات أعضاء هيئة التدريس الذين يعملون فى القسم الدراسى المحذوف. أما بالنسبة للمواد الدراسية التى يتم تنفيذها من قبل القسم الذى تم حذفه فسيتم حذفها أيضاً.

٤-١-١-٤-٥-٣ عملية التحديث (The Update Operation):

تستخدم عملية التحديث لتغيير قيم حقل أو أكثر فى سجل واحد أو أكثر من سجلات جدول ما. وكما هو الحال بالنسبة لعملية الإضافة، فإن عملية التحديث من الممكن أن تخترق أياً من القيود الأربعة التى تم استعراضها أعلاه. فمن الممكن أن تكون قيمة الحقل (أو الحقول) قيد التحديث تخرج عن مجال الحقل (أو الحقول). كذلك هو الحال بالنسبة لقيد تكامل الجدول، إذ إنه قد تتم محاولة تحديث سجل ما بحيث تكون قيمة مفتاحه الرئيسى متكررة مع سجل أخر فى نفس الجدول أو أن تكون قيمة المفتاح الرئيسى للسجل (أو جزء منه) قيمة غير معرفة؛ وبذلك يتم اختراق قيد تكامل الجدول. كذلك هو الحال بالنسبة لقيد القيم غير المعرفة، حيث إن قيمة أحد الحقول قد يتم تحديثها بحيث تكون غير معرفة على الرغم من أن القيد المفروض على الحقل الذى سيأخذ هذه القيمة لا يسمح بأن تكون القيمة غير معرفة مما يمثل اختراقاً لقيد القيم غير المعرفة. أما فيما يتعلق بقيد السلامة المرجعية، فإنه قد يتم اختراقه إذا تم تحديث قيمة حقل المفتاح الخارجى لأحد السجلات بحيث يشير إلى قيمة لا يوجد ما يقابلها من مفتاح رئيسى فى الجدول الذى من المفترض أن يشير إليه حقل المفتاح الخارجى، أو أن تكون قيمة المفتاح الخارجى غير معرفة بالرغم من أن المفتاح الخارجى قد تم توصيفه على أساس أنه لا يمكن أن يكون غير معرف. وفيما يلى بعض الأمثلة لمثل هذه الاختراقات وردود فعل نظام إدارة قاعدة البيانات إزاءها:

١- تحديث سجل عضو هيئة تدريس بحيث تكون قيمة مفتاحه الرئيسى (أو جزء منه) غير معرفة؛ سيقوم النظام برفض عملية التحديث هذه لاختراقها لقيد تكامل

الجدول الذى ينص على أن قيمة المفتاح الرئيسى لأى سجل لا يمكن أن تكون غير معرفة.

٢- تحديث سجل عضو هيئة تدريس بحيث تكون قيمة حقل مفتاحه الخارجى الجديدة التى تربطه بالقسم الدراسى الذى يعمل فيه عضو هيئة التدريس غير موجودة أصلاً ضمن المفاتيح الرئيسية لجدول الأقسام العلمية؛ سيقوم النظام برفض عملية التحديث هذه لاختراقها لقيد السلامة المرجعية.

٣- تحديث سجل عضو هيئة تدريس بحيث تكون قيمة مفتاحه الخارجى الذى يربطه بجدول الأقسام العلمية «غير معرفة» (NULL)؛ سيقوم النظام برفض عملية الإضافة هذه لاختراقها لقيد القيم غير المعرفة؛ إذ إن كل عضو هيئة تدريس يجب أن يرتبط بقسم يعمل فيه.

٤- تحديث سجل عضو هيئة تدريس بحيث تكون قيمة حقل رقم الهاتف مكونة من سلسلة حرفية ذات عشرة حروف؛ سيقوم النظام برفض عملية التحديث هذه لاختراقها لقيد المجال إذا كان حقل رقم الهاتف معرف على أساس كونه سلسلة حرفية بطول ثمانية حروف.

ولإمكان تحديث قيمة المفتاح الرئيسى لأى سجل فى قاعدة البيانات، فإن مثل هذا التحديث قد يؤدى إلى اختراق قيد السلامة المرجعية؛ وذلك لأنه بالإمكان أن توجد بعض السجلات فى جداول أخرى من جداول قاعدة البيانات تشير إلى المفتاح الرئيسى قبل تحديثه. ويمكن تصور عملية التحديث هذه وكأنها عملية حذف للسجل تتبعها عملية إضافة لنفس السجل ولكن بمفتاح رئيسى مختلف. لذا فإن الخروقات التى قد تسببها عملية التحديث هذه شبيهة بالخروقات التى تسببها عملية الحذف التى سبق إيضاحها أعلاه. لذلك فإن النموذج العلاقى يوفر أربعة خيارات عند حدوث خروقات لقيد السلامة المرجعية نتيجة لعمليات التحديث، شبيهة بتلك التى يوفرها لعمليات الحذف. ويتم استخدام الخيار المناسب من الخيارات الأربعة عند توصيف حقل المفتاح الخارجى أثناء إنشاء الجدول. وسيتم شرح طريقة توصيف هذه الخيارات باستخدام لغة الاستفسار البنائية فى الفصل السابع. أما هذه الخيارات الأربعة فهى كما يلى:

١- رفض عملية التحديث، وهذه الطريقة تسمى عملية «الرفض» (Reject) وهى الحالة الافتراضية عند اختراق قيد التكامل المرجعى نتيجة لعملية تحديث. فعلى سبيل المثال، لو تم تعريف قيد التكامل المرجعى لحقل المفتاح الخارجى فى جدول أعضاء

هيئة التدريس الذى يشير إلى جدول الأقسام العلمية على أنه من نوع الرفض، فإن تحديث قيمة المفتاح الرئيسى لأى سجل من سجلات جدول الأقسام العلمية سيترتب عليه رفض عملية التحديث إذا وجد أى سجل من سجلات أعضاء هيئة التدريس يشير إلى سجل جدول الأقسام العلمية المزمع تحديث قيمة مفتاحه الرئيسى.

٢- قبول عملية التحديث مع تحديث كافة حقول المفاتيح الخارجية فى كافة السجلات التى تشير للمفتاح الرئيسى قيد التحديث. وتسمى هذه الطريقة «التحديث المتسلسل» (Cascade). فعلى سبيل المثال، لو تم تعريف قيد التكامل المرجعى لحقل المفتاح الخارجى فى جدول أعضاء هيئة التدريس على أنه من نوع التحديث المتسلسل، فإن تحديث أى سجل من سجلات جدول الأقسام العلمية سيترتب عليه تحديث كافة سجلات أعضاء هيئة التدريس التابعين لهذا القسم الدراسى.

٣- قبول عملية التحديث مع وضع قيمة غير معرفة فى الحقول التى تشير للسجل الذى تم تحديث مفتاحه الرئيسى. وتسمى هذه الطريقة «وضع قيمة غير معرفة» (Set to NULL). فعلى سبيل المثال، لو تم تعريف قيد التكامل المرجعى لحقل المفتاح الخارجى فى جدول أعضاء هيئة التدريس على أنه بالإمكان أن يأخذ القيمة غير المعرفة، فإن تحديث المفتاح الرئيسى لأى سجل من سجلات جدول الأقسام العلمية سيترتب عليه وضع قيمة غير معرفة فى حقل المفتاح الخارجى لسجلات أعضاء هيئة التدريس الذين يتبعون للقسم الدراسى الذى تم تحديث قيمة مفتاحه الرئيسى. وكما هو الحال فى عملية الحذف، تجدر الإشارة إلى أنه فى مثل هذه الحالة لا يمكن توصيف حقل المفتاح الخارجى فى جدول أعضاء هيئة التدريس على أنه لا يمكن أن يكون «غير معرف» (Not NULL)؛ لأن خيار الحذف هذا سيترتب عليه خرق قيد القيم غير المعرفة.

٤- قبول عملية التحديث مع وضع «قيمة افتراضية» (Default Value) فى الحقول التى تشير للسجل الذى تم تحديث مفتاحه الرئيسى. وتسمى هذه الطريقة وضع قيمة افتراضية. فعلى سبيل المثال، لو تم تعريف قيد التكامل المرجعى لحقل المفتاح الخارجى فى جدول أعضاء هيئة التدريس على أنه من نوع وضع قيمة افتراضية، فإن تحديث قيمة المفتاح الرئيسى لأى سجل من سجلات الأقسام العلمية سيترتب عليه، وفق هذا الخيار، وضع القيمة الافتراضية المصاحبة لحقل المفتاح الخارجى فى كل سجل من سجلات أعضاء هيئة التدريس التابعين للقسم الدراسى الذى تم تحديث قيمة مفتاحه الرئيسى.

وتسمح نظم قواعد البيانات العلاقية بتوصيف ردود الفعل المناسبة إزاء خروقات قيود السلامة المرجعية، وذلك أثناء توصيف الحقول الخارجية لجداول قاعدة البيانات. كما تسمح هذه النظم بتوصيف ردود الفعل حسب نوع العملية التى أدت إلى خرق قيد من قيود السلامة المرجعية. فعلى سبيل المثال، عند تحديث حقل رمز القسم الدراسى قد يتم اختيار التحديث المتسلسل مما يعنى أن ينعكس رقم القسم الدراسى الجديد على كافة سجلات أعضاء هيئة التدريس. أما فى عملية الحذف، فإنه قد يتم اختيار الرفض مما يعنى عدم تنفيذ عملية الحذف عند وجود أعضاء هيئة تدريس يتبعون للقسم الدراسى المفترض حذف سجله من جدول الأقسام الدراسية. وسيتم إيضاح طريقة تعريف ردود الفعل المناسبة عند وجود قيود السلامة المرجعية أثناء شرح لغة الاستفسار البنائية فى الفصل السابع.

٤-٢ الجبر العلاقى (Relational Algebra):

يستعرض هذا الجزء من الفصل الجبر العلاقى الذى يعد إحدى «اللغات الرسمية» (Formal Languages) للنموذج العلاقى. فبالإضافة للمفاهيم التى تُمكن من تعريف هياكل البيانات والقيود المفروضة عليها، لابد أن يشتمل نموذج البيانات على مجموعة من العمليات التى تمكن من التعامل مع قاعدة البيانات التى تمت نمذجتها. وما الجبر العلاقى، إلا مجموعة من العمليات الأساسية التى تمكن من التعامل مع البيانات المخزنة وفق النموذج العلاقى بحيث تمكن من عملية استرجاع البيانات وفق المواصفات التى يطلبها المتعامل مع قاعدة البيانات. وتكون نتيجة الاسترجاع علاقة جديدة من الممكن أن تكون ناتجة من علاقة واحدة أو أكثر. وبذلك فإن العمليات الجبرية تنتج علاقات جديدة يمكن التعامل معها بإجراء المزيد من العمليات عليها وذلك من خلال استخدام نفس العمليات التى يوفرها الجبر العلاقى. وينتج عن سلسلة من العمليات الجبرية ما يعرف «بالتعبير الجبرى العلاقى» (Relational Algebra Expression) الذى تكون نتيجته علاقة جديدة تمثل ناتج الاسترجاع (أو الاستفسار) المنفذ على قاعدة البيانات.

ويستمد الجبر العلاقى أهميته من ثلاثة أبعاد رئيسية. الأول منهما يتمثل فى أن الجبر العلاقى يوفر أساساً رسمياً لإجراء العمليات على العلاقات فى النموذج العلاقى. أما الثانى فيتمثل فى كون الجبر العلاقى أساساً لبناء الاستفسارات (Queries) وزيادة فعاليتها (Optimization) فى نظم إدارة قواعد البيانات العلاقية (Relational Database Management Systems). أما البعد الثالث فيتمثل فى أن بعض

المفاهيـم الواردة فى الجبر العلاقى قد تم إدراجها ضمن اللغة القياسية للتعامل مع قواعد البيانات العلاقية والمعروفة باسم لغة الاستفسار البنائية (SQL). ولذلك فإن الجبر العلاقى يعد مكملاً لنموذج البيانات العلاقى.

ويمكن تصنيف عمليات الجبر العلاقى وفق فئتين رئيسيتين. فئة تحتوى على عمليات مستمدة من نظرية المجموعات الحسابية؛ وذلك لكون التعريف الرسمى للعلاقة فى النموذج العلاقى هو كونها مجموعة من الصفوف (أو السجلات) وبالتالى تنطبق عليها نظرية المجموعات الحسابية. وتتكون هذه الفئة من: عملية الاتحاد، وعملية التقاطع، وعملية الفرق. أما الفئة الثانية من العمليات الجبرية فتحتوى على عمليات قد تم تطويرها خصيصاً لقواعد البيانات العلاقية. وتتكون هذه الفئة من: عملية الاختيار (أو الاسترجاع)، وعملية الإسقاط، وعملية إعادة التسمية، وعملية الضرب الكرتيزى، وعملية الربط، وعملية القسمة.

وسيتم استعراض عمليات الاختيار، والإسقاط، وإعادة التسمية أولاً؛ وذلك لكونها عمليات أحادية تطبق على علاقة واحدة فقط. بعد ذلك سيتم استعراض العمليات الثنائية التى تطبق على علاقتين، عوضاً عن علاقة واحدة، وبحيث يتم استعراض العمليات المستمدة من نظرية المجموعات أولاً ومن ثم استعراض العمليات الثنائية الخاصة بالنموذج العلاقى.

٤-٢-١ العمليات الأحادية:

٤-٢-١-١ عملية الاختيار (Select):

تستخدم عملية الاختيار (SELECT) لاسترجاع مجموعة جزئية من صفوف جدول ما بحيث تتحقق عليها شروط محددة. ويمكن تصور عملية الاختيار على أنها مرشح للصفوف يبقى على الصفوف التى تحقق شرط الاختيار فى الجدول التى تطبق عليه العملية. كما يمكن تصورها وكأنها عملية تقسيم أفقى للجدول ينتج عنه مجموعتان من الصفوف: مجموعة يتحقق فيها شرط الاختيار ويتم اختيارها من خلال تنفيذ العملية، ومجموعة لا يتحقق فيها شرط الاختيار ويتم استبعادها من نتيجة العملية. فعلى سبيل المثال، لنفترض وجود العلاقة (R) التالية:

R		
A	B	C
a1	b1	c1
a1	b2	c2
a2	b3	c3
a3	b4	c4
a4	b5	c5
a5	b6	c6

ولنفترض أننا نرغب فى اختيار الصفوف التى يتحقق فيها شرط تساوى القيمة المخزنة فى الحقل (A) بالقيمة (a1). فى هذه الحالة تطبق عملية الاختيار كما يلى:

$$\sigma_{A = \text{“a1”}} (R)$$

وتكون نتيجة العملية جدولاً جديداً يحتوى على الصفوف التى تحتوى على القيمة (a1) فى الحقل (A) كما يلى:

$\sigma_{A = \text{“a1”}} (R)$		
A	B	C
a1	b1	c1
a1	b2	c2

وتعنى هذه النتيجة أن كلاً من الصف الأول والصف الثانى فى العلاقة (R) قد تم اختيارهما ضمن نتيجة العملية لانطباق شرط الاختيار وهو تساوى الحقل (A) بالقيمة (a1). فى حين لم يتم انطباق الشرط على بقية صفوف العلاقة، ولذلك لم يتم اختيارها ضمن نتيجة عملية الاختيار. ومثالاً تطبيقاً على قاعدة بيانات الجامعة الأهلية، لنفترض أننا نرغب فى معرفة أعضاء هيئة التدريس الذين يعملون فى قسم الحاسب الآلى ("CS") بالجامعة. ويمكن معرفة ذلك من خلال تطبيق عملية الاختيار على جدول أعضاء هيئة التدريس (FACULTY_T) بحيث يكون شرط الاختيار هو تساوى قيمة الحقل (Department_ID) بالقيمة ("CS") كما يلى:

$$\sigma_{\text{Department_ID} = \text{“CS”}} (\text{FACULTY_T})$$

فى هذه الحالة تكون نتيجة عملية الاختيار العلاقة التالية:

```
FACULTY_ FNAME        LNAME        PHONE_NO     SALARY DOB       DEPART
-------- ------------ ------------ -------- --------- --------- ------
310      Saleh        Aleesa       454-8932     30000 13-SEP-66 CS
320      Mohammed     Alhamad      454-5412     44000 13-MAY-65 CS
330      Ghanim       Alghanim     456-2234     44500 12-AUG-69 CS
340      Ibraheem     Alsaleh      454-1234     25000 20-JAN-70 CS
```

وبشكل عام تُمثل عملية الاختيار وفق الصيغة التالية:

$$\sigma_{\text{<Selection Condition>}} (R)$$

بحيث يستخدم الرمز (σ) لتمثيل عملية الاختيار ويقرأ سيجما (Sigma). أما شرط الاختيـــار (Selection Condition) فهو تعبيـــر ثنائى (Binary Operation) يطبق على حقول العلاقة (R) بحيث إما أن يتحقق الشرط وتكون قيمته صحيحة (True) أو أن لا يتحقق الشـــرط وتكون قيمته خاطئة (False). ويلاحظ أن تطبيق شرط الاختيار يتم على كل صف فى العلاقة على حدة بحيث يتم اختيار الصف ضمن نتيجة الاختيار عند تحقيقه لشـــرط الاختيار. أما فى حالة عدم تحقيقه لشـــرط الاختيار فلا يتم اختياره ضمن نتيجة عملية الاختيار. أما بالنسبة للعلاقة (R) فإنها بشكل عام تعبير جبرى علاقى. وفى أبســـط صورها عندما تكون اسماً لعلاقة موجودة أصلاً فى قاعدة البيانات دون إجراء أية عمليات جبرية عليها كما فى المثالين السابقين. وتكون نتيجة عملية الاختيار علاقة جديدة تحتوى على نفس حقول العلاقة التى تم تطبيق عملية الاختيار عليها.

إن التعبير الثنائى الموضح فى الشكل العام لتعليمة الاختيار يمكن أن يتم تكوينه من خلال مجموعة من التعابير البسيطة الموضحة بالشكل التالى:

<Attribute Name> <Comparison Operator> <Constant Value>

ففى المثال الســـابق كان اسم الحقل (Attribute Name) هو رمز القسم (Department_ ID) وعامل المقارنة هو المســـاواة (=) والقيمة الثابتة (Constant Value) هى أن يكون اسم القسم الحاسب الآلى ("CS"). كذلك يمكن أن تتم المقارنة بحقل آخر عوضاً عن قيمة ثابتة كما يوضح الشكل التالى:

<Attribute_1 Name> <Comparison Operator> <Attribute_2 Name>

ويقصد باســم الحقل (Attribute Name) اســم أحد الحقول فى العلاقة (R)، فى حين يقصد بعامل المقارنة (Comparison Operator) أحد عوامل المقارنة فى المجموعة {=, <, ≤, >, ≥, ≠}. أمــا القيمــة الثابتــة فيجب أن تكون من ضمــن القيم فى مدى (Domain) الحقل الذى سيطبق عليه شرط الاختيار. ويمكن استخدام عوامل الربط المنطقية {AND, OR, NOT} لتكوين شروط اختيار عامة. فعلى سبيل المثال لاختيار أعضاء هيئة التدريس الذين يعملون فى قســم الحاســب الآلــى وتزيد رواتبهم عن ٣٠,٠٠٠ أو يعملون فى قســم الرياضيات وتقل رواتبهم عن ٣٠,٠٠٠ فيمكن كتابة عملية الاختيار كما يلى:

$$\sigma_{(\text{Department_ID} = \text{"CS" AND Salary} > 30000) \text{ OR } (\text{Department_ID} = \text{"MATH" AND Salary} < 30000)} (\text{FACULTY_T})$$

وتكون نتيجة الاختيار كما يلى:

```
FACULTY_ FNAME        LNAME        PHONE_NO       SALARY DOB       DEPART
-------- ------------ ------------ -------- ---------- --------- ------
220      Fahad        Alhamid      456-7733      25900 07-OCT-70 MATH
320      Mohammed     Alhamad      454-5412      44000 13-MAY-65 CS
330      Ghanim       Alghanim     456-2234      44500 12-AUG-69 CS
```

وتطبــق عوامــل المقارنــة {=, <, ≤, >, ≥, ≠} على الحقول ذات المــدى المرتب القيم (Ordered Values) مثــل الأعداد والتواريخ والسلاســل الحرفية. أما إذا كانت الحقول ذات مدى غير مرتب القيم فلا ينطبق عليها من عوامل المقارنة ســوى عامل المســاواة وعامل عدم المســاواة: {=, ≠}. ومن أمثلة الحقول ذات المدى غير المرتب قيم الحقل «الجنس» الذى يستمد قيمه من إحدى القيمتين: ذكر {Male} أو أنثى {Female}.

ويتم تحديد نتيجة عملية الاختيار من خلال تطبيق شرط الاختيار على كل صف فى العلاقة (R) بحيث يتم اســتبدال قيمة الحقل مكان شرط الاختيار وعندما تكون نتيجة شــرط الاختيار صحيحــة (True) تتم عملية اختيار الصــف الذى تمت عليه عملية المقارنة. أما إذا كانت نتيجة شرط الاختيار خاطئة (False) فإنه لا يتم اختيار الصــف الذى تمت عليه عمليــة الاختيار. وتكون النتيجة النهائيــة لعملية الاختيار هى جميع الصفوف التى حققت شــرط الاختيار. أما بالنســبة للعمليات الشرطية

الثنائية {AND, OR, NOT} فلها نفس التفسيرات المعروفة فى الجبر الثنائى (Boolean Algebra) وهى كالتالى:

١- <Condition$_1$ AND Condition$_2$> يكون صحيحاً (True) إذا كان كل من الشـــرط الأول (Condition$_1$) والشرط الثانى (Condition$_2$) صحيحين (True).

٢- <Condition$_1$ OR Condition$_2$> يكـــون صحيحاً (True) إذا كان أى من الشـــرط الأول (Condition$_1$) أو الشرط الثانى (Condition$_2$) صحيحاً (True).

٣- <NOT Condition> يكون صحيحاً إذا كان الشرط (Condition) خاطئاً (False)، ويكون خاطئاً إذا كان الشرط (Condition) صحيحاً.

وتعتبـــر عملية الاختيار عملية أحادية (Unary Operation) لكونها تنطبق على علاقة واحـــدة فقـــط. كما أن عملية الاختيـــار تطبق على كل صف على حـــدة وبالتالى فإن شـــروط الاختيار لا يمكن أن تطبق على أكثر من صف فـــى العلاقة فى نفس الوقت. ويلاحظ أن درجة العلاقة التى تنتج من عملية الاختيار تكون مســـاوية لدرجة العلاقة الأصليـــة التى طبقت عليها عمليـــة الاختيار. ويعنى هذا أن عدد الحقول فى العلاقة الناتجة مســـاوٍ لعدد الحقول فى العلاقة الأصلية. ويلاحظ أيضاً أن عملية الاختيار عملية تبادلية بمعنى أن نتيجة تطبيقها المتسلســـل على علاقة ما لا تختلف باختلاف التسلســـل الذى طبقـــت فيه. ويعنى هذا أن تطبيق كل من المتسلســـلتين التاليتين من عملية الاختيار على العلاقة (R) متساوٍ.

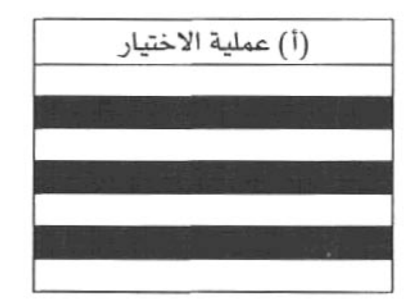
$$\sigma_{<Condition_1>} (\sigma_{<Condition_2>} (R)) = \sigma_{<Condition_2>} (\sigma_{<Condition_1>} (R))$$

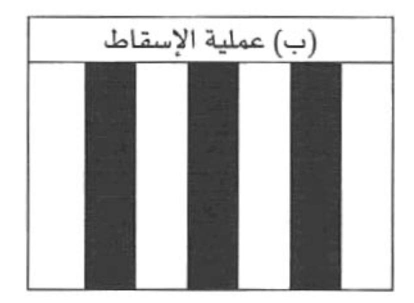
وبذلك فإنه يمكن تطبيق سلســـلة من عمليـــات الاختيار على علاقة ما بأى ترتيب كان دون إخلال بالنتيجة النهائية للسلســـلة. ونتيجة لذلك فإنه يمكن دمج سلسلة من عمليات الاختيار فى عملية اختيار واحدة باســـتخدام عامل الربط الثنائى (AND) كما يلى:

$$\sigma_{<Cond_1>} (\sigma_{<Cond_2>} (... (\sigma_{<Cond_n>} (R)) ...)) = \sigma_{<Cond_1> \text{ AND } <Cond_2> \text{ AND } ... \text{ AND } <Cond_n>} (R)$$

٤-٢-١-٢ عملية الإسقاط (Projection):

تســـتخدم عمليـــة الاختيار (SELECT) لاختيار تلك الصفــوف فى علاقة ما بحيث يتحقق فيها شرط الاختيار. على النقيض من ذلك فإن عملية الإسقاط تختار أعمدة معينـــة من العلاقة. ففى الوقت الذى تســـتخدم فيه عمليـــة الاختيار عند الرغبة فى اختيار بعض صفوف الجدول وفق شـــروط معينة عوضـــاً عن اختيارها جميعاً، والتى يمكن تصورها على أنها عملية تقسيم أفقى للعلاقة بحيث يتم تقسيمها إلى علاقتين: علاقة تحتوى على كافة الصفوف التى تحقق شـــرط الاختيار وهى التى تظهر ضمن نتيجة الاختيار، وعلاقة تحتوى على كافة الصفوف التى لا تحقق شـــرط الاختيار ولا تظهـــر من ضمن نتائج عملية الاختيار؛ فإن عملية الإســـقاط يمكن تصورها على أنها عملية تقســـيم رأســـى للعلاقة بحيث ينتـــج عنها علاقتان: علاقـــة تحتوى على كافة الأعمدة التى تم تحديدها ضمن عملية الإسقاط وهى التى تظهر ضمن نتيجة العملية، وعلاقـــة تحتـــوى على كافة الأعمدة التى لم يتم تحديدها ضمن عملية الإســـقاط ولا تظهر من ضمن نتائج العملية. والشكل التالى يوضح هذا التصور لكلتا العمليتين.





فعلى سبيل المثال، لنفترض وجود العلاقة (R) التالية:

R		
A	B	C
a1	b1	c1
a1	b1	c2
a2	b2	c2
a3	b3	c3
a4	b4	c4
a5	b5	c5

ولنفتــرض أننا نرغب فى إجراء عملية إســقاط على الحقــل (A) والحقل (B) من حقول العلاقة (R). فى هذه الحالة تطبق عملية الإسقاط كما يلى:

$$\pi_{A,\,B}\,(R)$$

وتكون نتيجة العملية العلاقة التالية:

$\pi_{A,\,B}\,(R)$	
A	B
a1	b1
a2	b2
a3	b3
a4	b4
a5	b5

ويلاحظ فى نتيجة عملية الإســقاط السابقة أن عدد صفوف العلاقة الناتجة منها قد أصبح خمســة صفوف عوضاً عن الصفوف الســتة التى تحتويها العلاقة الأصلية (R). ويعــزى الســبب وراء ذلك إلى أن عملية الإســقاط هى عمليــة جبرية علاقية تطبــق على علاقات ذات خصائص محددة وفقاً للنموذج العلاقى. وبذلك فإنها يجب أن تحافــظ على خصائص هذه العلاقات فى نتائجهــا. ومن ضمن هذه الخصائص أن العلاقــة هى مجموعة مــن الصفوف، وأن المجموعة، فى نظريــة المجموعات (Set Theory)، لا تحتوى على تكرارات ضمن عناصرها. لذلك فإن عملية الإسقاط، شأنها شــأن بقية عمليات الجبر العلاقى، يجب أن تحافظ على هذه الخاصية فى نتائجها. وحتى تحافظ على هذه الخاصية، فإن عملية الإسقاط تقوم بالإلغاء التلقائى للصفوف المتكــررة من نتيجتها النهائيــة. وحيث إن إلغاء الحقل (C) من العلاقة (R) بعد تطبيق عملية الإسقاط يترتب عليه تكرار فى نتيجة الصف الأول من العلاقة مع نتيجة الصف الثانى، فإن عملية الإسقاط قد قامت بإلغاء أحد الصفين من نتيجتها النهائية. وهذا هو السبب وراء حصولنا على خمسة صفوف عوضاً عن ستة صفوف فى نتيجة عملية الإسقاط.

ومثالاً تطبيقياً على قاعدة بيانات الجامعة الأهلية، لنفترض أننا نرغب فى معرفة الاسم الأول واسم العائلة والمرتب الشهرى لأعضاء هيئة التدريس فى الجامعة الأهلية. فى هذه الحالة نستخدم عملية الإسقاط كما يلى:

$$\pi_{FName,\,LName,\,Salary}\,(FACULTY_T)$$

وتكون نتيجة عملية الإسقاط هذه كما يلى:

FNAME	LNAME	SALARY
Khalid	Aloufi	35000
Fahad	Alhamid	25900
Saleh	Aleesa	30000
Mohammed	Alhamad	44000
Ghanim	Alghanim	44500
Ibraheem	Alsaleh	25000
Ahmad	Alotaibi	33900
Saleh	Alghamdi	44600
Yahya	Khorshid	36700
Salem	Alhamad	40000
Salman	Albassam	33800
Turki	Alturki	27800
Fahad	Alzaid	44300
Saud	Alkhalifa	44900
Mahmood	Alsalem	31900
Mishal	Almazid	29800
Sultan	Aljasir	43300
Ali	Albader	45300
Saad	Alzhrani	44200
Ahmad	Alsabti	33900

ويلاحظ فى نتيجة عملية الإســقاط الســابقة أنها تمت بشكل رأسى على الحقول التــى تم تحديدها ضمــن التعليمة مُلغية بذلــك بقية الحقول الموجــودة فى العلاقة الأصلية، وأن صفوف النتيجة لا تحتوى على أية تكرارات. أما الشــكل العام لتعليمة الإسقاط فهو كالتالى:

$$\pi_{<\text{Attribute List}>}\ (R)$$

ويمثل الرمز (π)، ويقرأ (باى)، تعليمة الإســقاط فى الجبــر العلاقى. أما قائمة الحقــول (Attribute List) فتمثل الحقول المراد إظهارها ضمن نتيجة عملية الإســقاط مــن قائمــة حقول العلاقة (R). ويلاحــظ، كما هو الحال فى حالــة عملية الاختيار، أن العلاقــة (R) عبارة عن تعبير علاقى نتيجتــه علاقة واحدة فقط، وأن هذا التعبير يكون فى أبســط صوره عندما تكون العلاقة هى إحدى العلاقات فى قاعدة البيانات. ونتيجة عملية الإسقاط هى علاقة تحتوى على الحقول المراد اختيارها فقط والمدرجة ضمن قائمة الحقول فى التعليمة. كما أن قائمة الحقول فى النتيجة تظهر مرتبة وفق ترتيبها فى عملية الإســقاط. وبهذا فإن درجة العلاقة الناتجة من تعليمة الإســقاط تكون مساوية لعدد الحقول فى قائمة حقول التعليمة.

إذا كانت الحقول المدرجة فى قائمة حقول التعليمة لا تحتوى على المفتاح الرئيسى للعلاقة (R) فإنه من المحتمل أن تظهر بعض الصفوف المتكررة فى نتيجة عملية الإسقاط، كما رأينا فى المثال السابق، إلا أن تعليمة الإسقاط تزيل هذه التكرارات عند حدوثها. لذا فإن نتيجة تعليمة الإسقاط هى مجموعة من الصفوف مما يعنى أن العلاقة الناتجة من عملية الإسقاط علاقة تتوافق فى مواصفاتها مع مواصفات النموذج العلاقى. تجدر الإشارة هنا إلى أن النموذج العلاقى الرسمى مبنى على نظرية المجموعات، كما أسلفنا سابقاً، وأن العلاقات يجب أن لا تحتوى على تكرارات ضمن صفوفها شأنها فى ذلك شأن المجموعات التى لا تحتوى على تكرارات ضمن عناصرها. على النقيض من ذلك فإن الحقائب فى علم الرياضيات تختلف عن المجموعات بكونها قد تحتوى على تكرارات ضمن عناصرها. ومثالاً تطبيقياً آخر على ذلك من قاعدة بيانات الجامعة الأهلية، لنفترض أننا نرغب فى معرفة رواتب (Salary) أعضاء هيئة التدريس فى الجامعة الأهلية، فإنه يمكن استخدام تعليمة الإسقاط التالية:

$$\pi_{Salary}\ (FACULTY_T)$$

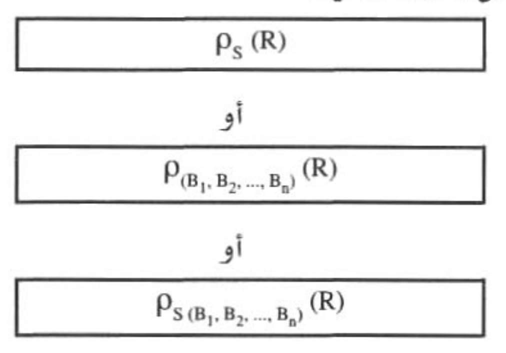
وتكون نتيجة تعليمة الإسقاط السابقة كما يلى:

```
SALARY
------
 25000
 25900
 27800
 29800
 30000
 31900
 33800
 33900
 35000
 36700
 40000
 43300
 44000
 44200
 44300
 44500
 44600
 44900
 45300
```

ويلاحظ فى النتيجة أنها تحتوى على تسعة عشر صفاً عوضاً عن عشرين (وهم عدد أعضاء هيئة التدريس فى الجامعة الأهلية). ويعزى السبب وراء ذلك إلى وجود عضوين من أعضاء هيئة التدريس يتقاضيان نفس الراتب، وهما : Ahmad Alotaibi و Ahmad Alsabti، وبالتالى تمت إزالة أحد الراتبين المتكررين وقدره (٣٣,٩٠٠) من نتيجة عملية الإسقاط. لذا فإن عدد الصفوف الناتجة من عملية الإسقاط هى علاقة تكون مساوية أو أقل فى عدد صفوفها من عدد الصفوف المدرجة فى العلاقة التى طبقت عليها عملية الإسقاط. إلا أن عدد الصفوف فى العلاقة الناتجة من عملية الإسقاط يكون مساوياً لعدد صفوف العلاقة التى طبقت عليها عملية الإسقاط فقط عندما تحتوى قائمة الحقول المدرجة ضمن حقول عملية الإسقاط على المفتاح الرئيسى للعلاقة التى طبقت عليها عملية الإسقاط. كما أن عملية الإسقاط، وعلى النقيض من عملية الاختيار، عملية غير تبادلية بمعنى أن نتيجة أى عمليتى إسقاط متعاقبتين على علاقة ما لا تكون متساوية عند اختلاف تسلسل تنفيذهما.

٤-٢-١-٣ عملية إعادة التسمية (Renaming):

عند تطبيق الجبر العلاقى على علاقات قاعدة البيانات تنتج علاقات وسيطة ليس لها أسماء معينة كما أن أسماء حقول هذه العلاقات الناتجة تكون بنفس أسماء حقول العلاقات الأصلية التى تم تطبيق الجبر العلاقى عليها. وللتحكم فى تسمية العلاقات والحقول المكونة لها يوفر الجبر العلاقى عملية إعادة التسمية. وتأخذ عملية إعادة التسمية أحد الأشكال الثلاثة التالية:



ويستخدم الرمز(ρ)، ويقرأ (روو)، لتمثيل عملية إعادة التسمية فى جميع الأشكال الثلاثة السابقة. وتستخدم العملية إما لإعادة تسمية علاقة ما، أو إعادة تسمية حقول العلاقة فقط، أو إعادة تسمية كل من اسم العلاقة وحقولها معاً. فالشكل الأول

للتعليمة يرمز لإعادة تسـمية العلاقة (R) تحت مسمى (S). وتكون نتيجة التعليمة هى كافة حقول العلاقة (R) ولكن تحت مسمى جديد للعلاقة باسم (S). أما الشكل الثانى فيرمز إلى إعادة تسـمية حقول العلاقة (R) فقط بمسـميات جديدة هى ($B_1, B_2, \ldots, B_n$) عوضاً عن مسـمياتها الأصلية. وتكون نتيجـة التعليمة فى هذه الحالة هى نفس العلاقة (R) ولكن بمسـميات حقول جديدة. ويلاحظ هنا أن إعادة تسـمية الحقول تتم بالترتيب من اليسـار إلى اليمين. أما الشـكل الثالث للتعليمـة فيرمز إلى إعادة تسمية كل من العلاقة (R) وحقولها معاً. وتكون نتيجة التعليمة فى هذه الحالة علاقة بمسـمى جديد هو (S) تحتوى على كافة حقول العلاقة (R) ولكن بمسـميات جديدة. تجدر الإشارة هنا إلى أن عملية إعادة التسمية لا تغير فى مسميات العلاقات الأصلية المكونة لقاعدة البيانات أو حقولها وإنما تقوم بإعادة تسمية هذه العلاقات أو الحقول تمهيداً لاستخدامها فى عمليات جبرية علاقية أخرى. وتعد هذه العملية مفيدة جداً خاصة عندما يحتوى التعبير الجبرى العلاقى على سلسلة طويلة من العمليات الجبرية العلاقية مما قد يسـتدعى تجزئته حتى يمكن التعامل معه بسـهولة. فلإظهار الاسم الأول واسـم العائلة والراتب لأعضاء هيئة التدريس الذين يعملون فى قسـم الحاسب الآلى ("CS") بالجامعة الاهلية، على سـبيل المثال، تطبق عملية اختيار وعملية إسقاط على علاقة أعضاء هيئة التدريس (FACULTY_T) كما يلى:

$$\pi_{FName,\ LName,\ Salary}\ (\sigma_{Department_ID=\text{"CS"}}\ (FACULTY_T))$$

ويوضح الشكل التالى ناتج هذا التعبير الجبرى العلاقى:

FNAME	LNAME	SALARY
Saleh	Aleesa	30000
Mohammed	Alhamad	44000
Ghanim	Alghanim	44500
Ibraheem	Alsaleh	25000

وبديلاً عن كتابة التعبير الجبرى العلاقى السابق الذى يحتوى على عمليتين جبريتين متداخلتين، يمكن كتابة التعبير على أنه سلسلة من عمليتين جبريتين علاقيتين، وذلك من خلال تسمية النتائج الوسيطة كما يلى:

$$CS_Employees \longleftarrow \sigma_{Department_ID=\text{"CS"}}\ (FACULTY_T)$$

$$Result \longleftarrow \pi_{FName,\ LName,\ Salary}\ (CS_Employees)$$

ويكون ناتج العملية الأولى العلاقة (CS_Employees) الموضحة كما يلى:

FACULTY_	FNAME	LNAME	PHONE_NO	SALARY	DOB	DEPART
310	Saleh	Aleesa	454-8932	30000	13-SEP-66	CS
320	Mohammed	Alhamad	454-5412	44000	13-MAY-65	CS
330	Ghanim	Alghanim	456-2234	44500	12-AUG-69	CS
340	Ibraheem	Alsaleh	454-1234	25000	20-JAN-70	CS

أما ناتج العملية الثانية فيكون العلاقة (Result) الموضحة كما يلى:

FNAME	LNAME	SALARY
Saleh	Aleesa	30000
Mohammed	Alhamad	44000
Ghanim	Alghanim	44500
Ibraheem	Alsaleh	25000

وعادة ما يكون تفكيك تعبير جبرى علاقى واحد مكوناً من سلسلة طويلة من العمليات الجبرية واستخراج نتائج وسيطة منها وصولاً للنتيجة النهائية أسهل فى التعامل من استخدام التعبير كوحدة واحدة. ويمكن استخدام عملية إعادة تسمية الحقول فى النتائج الوسيطة والنتائج النهائية. وتصبح عملية إعادة التسمية ذات أهمية كبيرة خاصة عند استخدام عمليات أكثر تعقيداً مثل عملية الاتحاد وعملية التقاطع اللتين سنتطرق لهما فى الجزء التالى.

٤-٢-٢ العمليات الثنائية:

٤-٢-٢-١ عمليات الجبر العلاقى الثنائية من نظرية المجموعات:

إن عملية الاختيار وعملية الإسقاط تمكناننا من الحصول على معلومات معينة من خلال تطبيقهما على علاقة واحدة فقط فى أى وقت من الأوقات؛ وذلك لكونهما عمليات أحادية لا تطبقان إلا على علاقة واحدة فقط. إلا أننا نحتاج فى الكثير من الأحيان إلى الحصول على معلومات لا يمكن الوصول إليها إلا من خلال عمليات تقوم بالدمج ما بين العلاقات. وتعد هذه العمليات عمليات ثنائية لكونها تتعامل (أو تطبق) على علاقتين فى نفس الوقت. ونستعرض فى هذا الجزء عمليات الجبر العلاقى الثنائية المستمدة من نظرية المجموعات. وهذه العمليات هى: عملية الاتحاد، وعملية التقاطع، وعملية الفرق.

٤-٢-٢-١-١ عملية الاتحاد (Union Operation):

ينتــج من تطبيق عملية الاتحــاد على علاقتين (R) و (S) علاقــة جديدة لها نفس بنيــة العلاقتين اللتين تم تطبيق عملية الاتحــاد عليهما. وتكون نتيجة العملية علاقة جديــدة تتكون من كافة الصفوف الموجودة فى العلاقتين ولكن دون تكرار. وتأتى هذه العملية متوافقة مع عملية الاتحاد فى نظرية المجموعات، إذ إن عملية الاتحاد بين أى مجموعتــين من العناصر تكون مجموعة جديــدة تحتوى على عناصر كلتا المجموعتين دون تكــرار فى عناصر المجموعة الناتجة. ولتطبيق عملية الاتحاد فى الجبر العلاقى فإنه يشــترط أن تكون كلتا العلاقتين قيــد تنفيذ عملية الاتحاد عليهما متوافقتين من حيــث عملية الاتحاد (Union-Compatible) بمعنى أن يكــون لكلتا العلاقتين نفس عدد الحقــول وأن يكون لكل زوج مــن الحقول (المتقابلين فى الترتيــب) فى كلتا العلاقتين من نفس المجال (Domain). ويكون عدد صفوف العلاقة الناتجة بحد أقصى مســاوياً لحاصل جمع عدد صفوف العلاقتين. إلا أن هذا العدد يتناقص فى حالة وجود تكرار مــا بين صفوف العلاقتين لأن كل صف متكرر ما بــين العلاقتين لا يظهر فى النتيجة إلا مــرة واحدة فقط. وتمثل عملية الاتحاد بين علاقتين (R) و (S) كما هو موضح فى الشكل التالى:


$R \cup S$

حيث إن الرمز (∪) يستخدم لتمثيل عملية الاتحاد. ويلاحظ أن عملية التوافق من حيــث الاتحاد لا تنص علــى أن يحمل كل زوج من الحقول (المتقابلين فى الترتيب) فى العلاقتين نفس المســمى؛ وذلك لأنه باستطاعتنا دائماً أن نعيد تسمية الحقول للتوافق مع بعضها من حيث المســميات. ومن الأمثلة التطبيقية لعملية الاتحاد لنفترض وجود العلاقتــين (R) و(S) المتوافقتين من حيث الاتحــاد وأننا نرغب فى إجراء عملية اتحاد بينهما. فستكون نتيجة عملية الاتحاد ما بين العلاقتين كما يلى:

R

A	B
a1	b1
a2	b2
a3	b3
a4	b4

∪

S

A	B
a1	b1
a2	b2
a3	b1

⇒

R ∪ S

A	B
a1	b1
a2	b2
a3	b3
a3	b1
a4	b4

ويلاحـــظ فى العلاقة الناتجة مـــن عملية الاتحاد احتواؤها على خمســـة صفوف عوضاً عن الصفوف السبعة التى تحتويها العلاقتان (R) و(S) وذلك لتكرار الصف الأول والصف الثانى فى كلتا العلاقتين. وبالتالى فإن عدد الصفوف فى ناتج عملية الاتحاد بـــين العلاقتـــين أقل من حاصل جمع عدد صفوفهما. كمـــا يلاحظ أن الصف الثالث فى العلاقـــة (R) والصف الثالث فى العلاقة (S) قد تم إدراجهما ضمن نتيجة العملية على الرغم من وجود تطابق جزئى بينهما، حيث إن القيمة المخزنة فى الحقل (A) فى كلا الصفين متساوية. ويعنى هذا أن أى صفين فى علاقتين ما يعدان متساويين عند تســـاوى كافة القيم المخزنة فى كافة حقولهمـــا فقط. لذلك لا يعد الصف الثالث فى العلاقـــة (R) والصف الثالث فى العلاقة (S) متطابقين. وبذلك يجب إدراجهما ضمن نتيجة عملية الاتحاد.

ومن الأمثلـــة التطبيقية على قاعدة بيانات الجامعـــة الأهلية، لنفترض أننا نرغب فى معرفة الاســـم الأول واســـم العائلة لكافة أعضاء هيئة التدريس وكافة الطلبة فى الجامعـــة. فى هذه الحالة نطبق عملية إســـقاط على علاقـــة أعضاء هيئة التدريس لاختيار حقول الاســـم الأول واســـم العائلة. ونطبق عملية إسقاط أخرى مماثلة على علاقة الطلبة. وتكون نتائج العمليتين فى هذه الحالة متوافقة من حيث عملية الاتحاد لكون كلتا العلاقتين الوسيطتين الناتجتين لهما نفس عدد الحقول (وهما حقلان يمثلان الاســـم الأول واســـم العائلة)، وأن كل زوج من الحقول المتقابلة لهما نفس مدى القيم حيث إنهما معرفان على أســـاس أنهما حقول حرفية بطـــول (١٢) حرفاً (CHAR(12)) فى قاعدة بيانات أوراكل. أما شـــكل العمليات المطبقة للحصول على النتيجة المطلوبة فهو كما يلى:

$$(\pi_{FName,\ LName}\ (FACULTY_T)) \cup (\pi_{FName,\ LName}\ (STUDENT_T))$$

ويمكـــن اعتبار عملية الاتحاد عملية متعددة بمعنى أنه يمكن تطبيقها على أى عدد من العلاقات يفوق الاثنتين ضمن سلســـلة من عمليات الاتحاد، وأنها عملية مشـــاركة (Associative Operation)، بمعنـــى أن ترتيـــب تطبيقهـــا على عدد مـــن العلاقات ضمن سلســـلة من عمليات الاتحاد لا يؤثـــر فى النتيجة النهائية لسلســـلة عمليات الاتحاد. ويمكن تمثيل هذه الخاصية كما يلى:

$$R \cup (S \cup T) = (R \cup S) \cup T$$

فى سلسلة عمليات الاتحاد السابقة تكون نتيجة تطبيق العملية على العلاقتين (S) و(T) ومــن ثم تطبيق عملية الاتحاد على العلاقة الناتجة والعلاقة (R) (كما هو موضح فى الجزء الأيسر من المعادلة) مساويةً لنتيجة تطبيق عملية الاتحاد على العلاقتين (R) و (S) ومن ثم تطبيق عملية الاتحاد على العلاقة الناتجة والعلاقة (T) (كما هو موضح فى الجزء الأيمن من المعادلة).

٤-٢-٢-١-٢ عملية التقاطع (Intersection Operation):

ينتــج من تطبيق عملية التقاطع على علاقتين (R) و (S)، وكما هو الحال فى عملية الاتحاد، علاقة جديدة لها نفس بنية العلاقتين اللتين تم تطبيق عملية التقاطع عليهما. إلا أن ناتج عملية التقاطع، وعلى خلاف عملية الاتحاد، يتكون من الصفوف المشــتركة فى العلاقتين بمعنى أن كل صف من صفوف نتيجة العملية يجب أن يكون موجوداً فى كلتا العلاقتين. وتأتى هذه العملية متوافقة مع عملية التقاطع فى نظرية المجموعات، حيث إن عملية التقاطع بين أية مجموعتين من العناصر تكون مجموعة جديدة تحتوى على العناصر المشــتركة بين عناصــر المجموعتين. ولتطبيق عملية التقاطع فى الجبر العلاقى فإنه يشترط، كما هو الحال فى عملية الاتحاد، أن تكون العلاقتان متوافقتين مــن حيث عملية الاتحــاد (Union-Compatible). ويعنى هــذا أن يكون للعلاقتين نفس عدد الحقول وأن يكون لكل زوج من الحقول (المتقابلين فى الترتيب) فى كلتا العلاقتين نفــس المجــال (Domain). وتمثل عملية التقاطع بين علاقتين (R) و (S) كما هو موضح فى الشكل التالى:



R ∩ S

حيث إن الرمز (∩) يستخدم لتمثيل عملية التقاطع. ومن الأمثلة التطبيقية لعملية التقاطــع، لنفترض وجود العلاقتين (R) و (S) المتوافقتين من حيث الاتحاد وأننا نرغب فــى إجراء عملية تقاطع بينهما. فســتكون نتيجة عمليــة التقاطع بين العلاقتين كما يلى:

R

A	B
a1	b1
a2	b2
a3	b3
a4	b4

∩

S

A	B
a1	b1
a2	b2
a3	b1

⇒

R ∩ S

A	B
a1	b1
a2	b2

ويلاحظ فى نتيجة عملية التقاطع السابقة بين العلاقتين (R) و(S) أنها تحتوى على الصفوف المشتركة بين العلاقتين وهما الصف الأول والصف الثانى. كما يلاحظ عدم إدراج أى من الصف الثالث فى العلاقة (R) أو الصف الثالث فى العلاقة (S) ضمن نتيجة العملية على الرغم من وجود تطابق جزئى بينهما حيث إن القيمة المخزنة فى الحقل (A) فى كلا الصفين متساوية. ويعنى هذا أن أى صفين فى علاقتين ما يعدان متساويين فقط فى حال تساوت كافة القيم المخزنة فى كافة حقولهما. لذلك لا يعد الصف الثالث فى العلاقة (R) والصف الثالث فى العلاقة (S) متطابقين. وبذلك يجب عدم إدراج أى منهما ضمن نتيجة عملية التقاطع.

ومن الأمثلة التطبيقية على قاعدة بيانات الجامعة الأهلية، لنفترض وجود بعض أعضاء هيئة التدريس الذين سبق أن كانوا من طلبة الجامعة قبل تعيينهم فيها أعضاءً لهيئة التدريس، وأننا نرغب فى معرفة هؤلاء أعضاء هيئة التدريس من خلال إظهار الاسم الأول واسم العائلة لكل منهم. فى هذه الحالة نطبق عملية إسقاط على علاقة أعضاء هيئة التدريس لاختيار حقول الاسم الأول واسم العائلة. ونطبق عملية إسقاط أخرى مماثلة على علاقة الطلبة. وتكون نتائج العمليتين فى هذه الحالة متوافقة من حيث عملية الاتحاد لكون كلتا العلاقتين الوسيطتين الناتجتين لهما نفس عدد الحقول (وهما حقلان يمثلان الاسم الأول واسم العائلة)، وأن كل زوج من الحقول المتقابلة لهما نفس مدى القيم حيث إنهما معرفان على أساس أنهما حقول حرفية بطول (١٢) حرفاً (CHAR(12)) فى قاعدة بيانات أوراكل. أما شكل العمليات المطبقة للحصول على النتيجة المطلوبة فهو كما يلى:

$$(\pi_{FName,\ LName}\ (FACULTY_T)) \cap (\pi_{FName,\ LName}\ (STUDENT_T))$$

ويمكن اعتبار عملية التقاطع، وكما هو الحال فى عملية الاتحاد، عملية متعددة بمعنى أنه يمكن تطبيقها على أى عدد من العلاقات يفوق الاثنتين ضمن سلسلة من عمليات التقاطع، وأنها عملية مشاركة (Associative Operation) بمعنى أن ترتيب تطبيقها على عدد من العلاقات ضمن سلسلة من عمليات التقاطع لا يؤثر فى النتيجة النهائية لسلسلة عمليات التقاطع. ويمكن تمثيل هذه الخاصية كما يلى:

$$R \cap (S \cap T) = (R \cap S) \cap T$$

فى سلسلة عمليات التقاطع السابقة تكون نتيجة تطبيق العملية على العلاقتين (S) و(T) ومن ثم تطبيق عملية التقاطع على العلاقة الناتجة و العلاقة (R) (كما هو موضح

فى الجزء الأيسر من المعادلة) مساويةً لنتيجة تطبيق عملية التقاطع على العلاقتين (R) و (S) ومن ثم تطبيق عملية التقاطع على العلاقة الناتجة والعلاقة (T) (كما هو موضح فى الجزء الأيمن من المعادلة).

٤-٢-٢-١-٣ عملية الفرق (Minus Operation):

ينتـج من تطبيق عملية الفـرق على العلاقتين (R) و (S)، وكما هو الحال فى عملية الاتحاد وعملية التقاطع، علاقة جديدة لها نفس بنية العلاقتين اللتين تم تطبيق عملية الفرق عليهما. إلا أن ناتج عملية الفرق، وعلى خلاف عملية الاتحاد وعملية التقاطع، يتكـون من كافة الصفوف الموجـودة فى العلاقة الأولى وليسـت موجودة فى العلاقة الثانيـة. وتأتى هذه العملية متوافقة مع عملية الفرق فى نظرية المجموعات، حيث إن عمليـة الفرق بين أى مجموعتين من العناصر تكون مجموعة جديدة تحتوى على كافة عناصر المجموعة الأولى وليسـت موجودة ضمـن عناصر المجموعة الثانية. ولتطبيق عملية الفرق فى الجبر العلاقى فإنه يشترط، كما هو الحال فى عملية الاتحاد وعملية التقاطـع، أن تكون كلتا العلاقتين قيد تنفيـذ عملية الفرق عليهما أن يكونا متوافقتين مـن حيث عملية الاتحـاد (Union-Compatible). ويعنى هـذا أن يكون لكلتا العلاقتين نفـس عدد الحقـول وأن يكون لكل زوج من الحقول (المتقابلين فـى الترتيب) فى كلتا العلاقتين نفس مدى (Domain) القيم التى يمكن أن يحتويا عليها. وتمثل عملية الفرق بين علاقتين (R) و (S) كما هو موضح فى الشكل التالى:

R – S

حيث إن الرمز (-) يسـتخدم لتمثيل عملية الفـرق. ومن الأمثلة التطبيقية لعملية الفرق، لنفترض وجود العلاقتين (R) و (S) المتوافقتين من حيث الاتحاد وأننا نرغب فى إجراء عملية فرق بينهما. فستكون نتيجة عملية الفرق بين العلاقتين كما يلى:

R	
A	B
a1	b1
a2	b2
a3	b3
a4	b4

–

S	
A	B
a1	b1
a2	b2
a3	b1

⇒

R – S	
A	B
a3	b3
a4	b4

ويلاحظ فى نتيجة عملية الفرق السابقة بين العلاقتين (R) و(S) أنها تحتوى على الصفوف الموجودة فى العلاقة (R) وليست موجودة فى العلاقة (S). وفى هذه الحالة هما الصفان الثالث والرابع من صفوف العلاقة (R).

ومن الأمثلة التطبيقية على قاعدة بيانات الجامعة الأهلية، لنفترض مرة أخرى وجود بعض أعضاء هيئة التدريس الذين سبق أن كانوا من طلبة الجامعة قبل تعيينهم فيها أعضاءً لهيئة التدريس، وأننا نرغب فى معرفة أعضاء هيئة التدريس الذين لم يسبق لهم أن كانوا من طلبة الجامعة، وذلك من خلال إظهار الاسم الأول واسم العائلة لكل منهم. فى هذه الحالة نطبق عملية إسقاط على علاقة أعضاء هيئة التدريس لاختيار حقول الاسم الأول واسم العائلة. ونطبق عملية إسقاط أخرى مماثلة على علاقة الطلبة. وتكون نتائج العمليتين فى هذه الحالة متوافقة من حيث عملية الاتحاد لكون كلتا العلاقتين الوسيطتين الناتجتين لهما نفس عدد الحقول (وهما حقلان يمثلان الاسم الأول واسم العائلة)، وأن كل زوج من الحقول المتقابلة لهما نفس مدى القيم حيث إنهما معرفان على أساس أنهما حقول حرفية بطول (١٢) حرفاً ((CHAR(12)) فى قاعدة بيانات أوراكل. أما شكل العمليات المطبقة للحصول على النتيجة المطلوبة فهو كما يلى:

$$(\pi_{FName,\ LName}\ (FACULTY_T)) - (\pi_{FName,\ LName}\ (STUDENT_T))$$

ويلاحظ على عملية الفرق، وبشكل عام، أنها عملية غير تبادلية بمعنى أن نتيجة تطبيقها المتسلسل يختلف باختلاف التسلسل الذى طبقت فيه. ويمكن توضيح ذلك كما يلى:

$$R - S \neq S - R$$

ولإيضاح هذه الخاصية نذكر مثالاً تطبيقياً على قاعدة بيانات الجامعة الأهلية: لنفترض أننا طبقنا عملية الفرق بين نتيجة عملية الإسقاط على علاقة أعضاء هيئة التدريس ونتيجة علاقة الإسقاط على علاقة الطلبة، فى المثال السابق، بشكل معاكس كما هو موضح فيما يلى:

$$(\pi_{FName,\ LName}\ (STUDENT_T)) - (\pi_{FName,\ LName}\ (FACULTY_T))$$

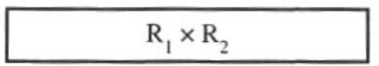
فـــى هذه الحالة لن تكون نتيجة تطبيق عمليـــة الفرق ممثلة للمطلوب وهو «إظهار أســـماء أعضاء هيئة التدريس الذين ســـبق أن كانوا من طلبة الجامعة». وإنما ستكون النتيجة «أســـماء الطلبة الذين ليسوا من أعضاء هيئة التدريس». وقد تكون مثل هذه النتيجة غير ذات مدلولات منطقية إذا ما اعتبرنا أن عضو هيئة التدريس لا يمكن أن يكون طالباً فى الجامعة!

٤-٢-٢-٢ عمليات الجبر العلاقى الثنائية الخاصة بالنموذج العلاقى:

يســـتعرض هذا الجزء عمليات الجبر العلاقى الثنائية الخاصة بالنموذج العلاقى. وهذه العمليات هى: عملية الضرب الكرتيزى، وعملية الربط، وعملية القسمة.

٤-٢-٢-٢-١ عملية الضرب الكرتيزى (Cartesiân Product):

عملية الضـــرب الكرتيزى هى عملية تطبق على المجموعات وهى عملية ثنائية بين معطيين يكون كل منهما مجموعة من الصفوف تتمثل فى علاقة ما. وتســـتخدم هذه العملية لدمج صفوف علاقتين بشـــكل توليفى بمعنى إظهار جميع الاحتمالات الممكنة للدمـــج بـــين علاقتين. وتكون نتيجة العملية التى يرمز لهـــا برمز علامة الضرب (×) علاقـــة جديدة ذات درجة تســـاوى حاصل جمع درجة العلاقتـــين اللتين تمت عليهما عملية الضرب الكرتيزى. ونعنى بدرجة العلاقة هنا، كما أســـلفنا سابقاً، عدد حقول العلاقة. كما يكون عدد صفوف العلاقة الناتجة مساوياً لعدد صفوف العلاقة الأولى مضروباً فى عدد صفوف العلاقة الثانية. وتمثل عملية الضرب الكرتيزى بين علاقتين (R_1) و (R_2) كما هو موضح فى الشكل التالى:



$$R_1 \times R_2$$

ومن الأمثلة التطبيقية لعملية الضرب الكرتيزى، لنفترض وجود العلاقتين (R_1) ذات الدرجة (٣) وحقولها هى (A, B, C) والعلاقة (R_2) ذات الدرجة (٢) وحقولها هى (Y, Z)، وأننا نرغب فى إجراء عملية الضرب الكرتيزى عليهما. عندئذ ســـتكون نتيجة عملية الضرب الكرتيزى بين العلاقتين كما هو موضح بالشكل التالى:

R_1		
A	B	C
a1	b1	c1
a2	b2	c2
a3	b3	c3

×

R_2	
Y	Z
y1	z1
y2	z2

⇒

$R_1 \times R_2$				
A	B	C	Y	Z
a1	b1	c1	y1	z1
a1	b1	c1	y2	z2
a2	b2	c2	y1	z1
a2	b2	c2	y2	z2
a3	b3	c3	y1	z1
a3	b3	c3	y2	z2

ويلاحظ أن درجة العلاقة الناتجة أصبحت (٥) وحقولها هى حقول العلاقة (R_1) مدموجة مع حقول العلاقة (R_2). أما نتيجة العملية فهى دمج للصف الأول من العلاقة (R_1) مع كل صف من صفوف العلاقة (R_2)، ودمج للصف الثانى من العلاقة (R_1) مع كل صف من صفوف العلاقة (R_2)، ودمج للصف الثالث من العلاقة (R_1) مع كل صف من صفوف العلاقة (R_2). ويعنى هذا أن نتيجة الضرب الكرتيزى ما هى إلا استعراض لكافة توليفات صفوف علاقتين بحيث ينتج عن هذه التوليفات علاقة جديدة درجتها هى حاصل جمع درجتى العلاقتين، وهى عدد حقول العلاقة الأولى مجموعة على عدد حقول العلاقة الثانية، اللتين تمت عليهما عملية الضرب الكرتيزى. ويكون عدد صفوفها هو حاصل ضرب عدد صفوف العلاقة الأولى فى عدد صفوف العلاقة الثانية. ففى مثالنا المستخدم أصبح عدد صفوف العلاقة الناتجة هو (٦) والذى يمثل حاصل ضرب عدد صفوف (R_1)، وهو (٣)، بعدد صفوف (R_2)، وهو (٢). وتجدر الإشارة هنا إلى أنه لا يشترط فى الضرب الكرتيزى توافق العلاقتين قيد تنفيذ إجراء العملية عليهما من حيث الاتحاد كما هو الحال فى عمليات الاتحاد، والتقاطع، والفرق.

ولأنه من الممكن أن تحتوى علاقتان على نفس مسميات الحقول، فإنه يتم إضافة مسمى العلاقة للحقول المتشابهة فى نتيجة عملية الضرب الكرتيزى حتى تتم المحافظة على خاصية تفرد مسميات الحقول فى العلاقة الناتجة. كما أنه يمكن استخدام عملية إعادة التسمية مع إحدى العلاقتين لإعادة تسمية حقولها المتشابهة فى المسمى مع العلاقة الأخرى قبل إجراء عملية الضرب الكرتيزى عليهما.

ومن الأمثلة التطبيقية على قاعدة بيانات الجامعة الأهلية، لنفترض أننا نرغب فى إظهار أسماء أعضاء هيئة التدريس (الاسم الأول واسم العائلة) مقروناً بأسماء الأقسام التى يعملون فيها. فى هذه الحالة يتم تنفيذ التعبير الجبرى العلاقى التالى:

$$\pi_{FName,\ LName,\ Department_ID}\ (\sigma_{Department_ID\ =\ FDEPT_ID}\ (\rho_{Department_ID\ as\ FDEPT_ID}\ (FACULTY_T) \times DEPARTMENT_T))$$

وتنفـذ العمليـات فى التعبير الجبرى العلاقى السـابق وفق أولويـات متوافقة مع الأقواس المستخدمة، وذلك من الداخل للخارج بحيث تنفذ عملية إعادة التسمية لحقل رمز القسـم (Department_ID) فى علاقة أعضاء هيئة التدريس أولاً ليصبح (FDEPT_ID). ويعزى السـبب وراء إعادة تسـمية هذا الحقل إلى تطابق مسماه مع مسمى حقل آخر فى علاقة الأقسـام العلميـة (DEPARTMENT_T). بعد ذلك تنفذ عملية الضرب الكرتيزى الذى تكون نتيجته كافة التوليفات الممكنة ما بين صفوف علاقة أعضاء هيئة التدريس مع صفوف علاقة الأقسام العلمية. ونظراً لأننا قد قمنا بإعادة تسمية الحقل الممثل لرمز القسم فى علاقة أعضاء هيئة التدريس، فإن العلاقة الناتجة لا تحتوى على أية ازدواجية فى مسـميات حقولها وبالتالى فإنها علاقة صحيحة تتوافق مع شـروط علاقـات النموذج العلاقى. بعد ذلك تنفذ عملية الاختيار التى تقوم باختيار الصفوف التى تتساوى فيها حقول رمز القسم الذى يعمل فيه عضو هيئة التدريس مع رمز القسم المـدون فى قائمة الأقسـام العلمية. وتكون النتيجة الوسـيطة حتـى هذه المرحلة من التعبير علاقة تحتوى على صفوف مكونة من كافة حقول علاقة أعضاء هيئة التدريس مربوطة بحقول الأقسام الدراسية التى يعملون فيها. وأخيراً تنفذ عملية الإسقاط التى تقوم بإظهار الحقول الثلاثة المطلوبة، وهى: الاسـم الأول لعضو هيئة التدريس، واسم عائلته، واسم القسم الذى يتبعه عضو هيئة التدريس.

٤-٢-٢-٢-٢ عملية الربط (Join Operation):

تسـتخدم عملية الربط (Join) لدمج سـجلات تربط بينهمـا علاقة ما بحيث تتبع هذه السـجلات لجدولين مختلفين. ويرمز لعملية الربط بالرمز (⋈). وتعد عملية الربط واحدة من أهم العمليات فى النموذج العلاقى لكونها تمكن من معالجة العلاقات التـى تربط بين الجداول المختلفة فى قاعدة البيانـات. ولإيضاح ذلك، لنفترض أننا نرغب فى معرفة أسماء الطلبة وأرقام المواد الدراسية التى سجلوا فيها ونتائجهم فى هذه المواد. فى هذه الحالة يتم ربط جدول التسـجيل (ENROLLMENT_T) مع جدول الطلبة (STUDENT_T) للحصول على كافة بيانات الطلبة والمواد التى قاموا بالتسجيل فيها. بعد ذلك تُتبع عملية الربط بعملية إسقاط على الحقول المطلوبة، كما يلى:

$$Result \longleftarrow \pi_{FName,\ LName\ Course_ID,\ Grade}\ (STUDENT_T \bowtie ENROLLMENT_T)$$

وتســمى العملية السابقة عملية ربط التســاوى؛ إذ إن عملية الربط تتم بين جدول الطلبة وجدول التسجيل وفق الحقول المشتركة بين الجدولين (وهى حقل «رقم الطالب» فى مثالنا). ويمكن كتابة عملية ربط التساوى (Equi-Join) بشكل صريح كما يلى:

$$STUDENT_T \bowtie_{STUDENT_T.Student_ID\ =\ ENROLLMENT_T.Student_ID} ENROLLMENT_T$$

وتكون نتيجة العملية السابقة، كما يلى:

FNAME	LNAME	COURSE_	GRADE
Saleh	Alhamad	CHEM101	4
Abdullah	Aloufi	CHEM101	3
Khalid	Alsultan	CHEM101	4
Salem	Algamdi	CHEM101	3
Mishal	Alyousef	CHEM101	1
Saleh	Alhamad	CS101	2
Mishal	Alyousef	CS101	4
Saleh	Alhamad	CS102	3
Mishal	Alyousef	CS102	4
Saleh	Alhamad	ENGL101	3
Abdullah	Aloufi	ENGL101	4
Salem	Algamdi	ENGL101	4
Mishal	Alyousef	ENGL101	4
Saleh	Alhamad	ENGL102	1
Mishal	Alyousef	ENGL102	4
Saleh	Alhamad	MATH101	3
Abdullah	Aloufi	MATH101	2
Salem	Algamdi	MATH101	0
Mishal	Alyousef	MATH101	2
Saleh	Alhamad	MATH102	2
Mishal	Alyousef	MATH102	0
Saleh	Alhamad	STAT101	2
Mishal	Alyousef	STAT101	3

كمــا أنــه ليس من الضرورى أن تتم عملية الربط وفق عملية التســاوى فحســب، ولكــن يمكن إجــراء عملية الربط وفق أى من عوامل المقارنــة التالية: $\{>, \geq, <, \leq, \neq\}$ بالإضافــة لعامل التســاوى. كما يوجد أنواع مختلفة مــن عمليات الربط من ضمنها

الربط الطبيعى، الذى يلغى أحد الحقول المتكررة من النتيجة عند تساوى مسميات حقول الربط، والربط الخارجى، الذى يكون من ضمن نتائجه تلك السجلات الموجودة فى أحد الجدولين والتى لا يوجد ما يقابلها فى الجدول الآخر. وسيتم التطرق للأنواع المختلفة من عمليات الربط أثناء شرح لغة الاستفسار البنائية (فى الفصل الثامن).

٤-٢-٢-٢-٣ عملية القسمة (Division Operation):

تستخدم عملية القسمة فى بعض الحالات الخاصة من الاستفسارات. ويرمز لعملية القسمة بالرمز (÷). فعلى سبيل المثال، إذا افترضنا وجود جدولين هما (R) و(S)، وأردنا معرفة الجدول (T) الذى يمثل (R÷S) فإن النتيجة ستكون كما يلى:

R	
X	Y
x1	y1
x2	y1
x3	y1
x4	y1
x1	y2
x3	y2
x2	y3
x3	y3
x4	y3
x1	y4
x2	y4
x3	y4

÷

S
Y
x1
x2
x3

$\Rightarrow$

T = R ÷ S
Y
y1
y4

وتعنى النتيجة السابقة أن عملية القسمة تكافئ العمليات الجبرية الثلاث التالية:

$$T_1 \longleftarrow \pi_Y (R)$$
$$T_2 \longleftarrow \pi_Y ((S \times T_1) - R)$$
$$T \longleftarrow T_1 - T_2$$

وبمعنى آخر فإن ناتج عملية القسمة هو جدول (T) يتكون من مجموعة من السجلات بحيث يكون كل ســجل فى الجدول مدمجاً مع **كافة** ســجلات الجدول (S)، الذى يمثل مقام عملية القسمة، من ضمن سجلات الجدول (R). كما تجدر الملاحظة بأن حقول الجدول (S)، ولتكن $A=\{a_1, a_2, ..., a_n\}$، يجب أن تكون مجموعة جزئية من حقول الجدول (R)، ولتكن $B=\{b_1, b_2, ..., b_m\}$، أى $(B \subseteq A)$.

وعلــى افتراض وجود الجدول «موظف» والجدول «مشــروع» التاليين، وأننا نرغب فى معرفة الموظفين الذين يعملون على كافة المشــاريع؛ فإنه يمكن إجراء عملية قسمة للحصول على النتيجة المطلوبة، كما يلى:

EMP

ENO	PNO	PName	Budget
E1	P1	Instrumentation	150000
E2	P1	Instrumentation	150000
E2	P2	Database Develop.	135000
E3	P1	Instrumentation	150000
E3	P4	Maintenance	310000
E4	P2	Instrumentation	150000
E5	P2	Instrumentation	150000
E6	P4	Maintenance	310000
E7	P3	CAD/CAM	250000
E8	P3	CAD/CAM	250000
E3	P2	Database Develop.	135000
E3	P3	CAD/CAM	250000

÷

PROJ

PNO	PName	Budget
P1	Instrumentation	150000
P2	Database Develop.	135000
P3	CAD/CAM	250000
P4	Maintenance	310000

⟹

EMP ÷ PROJ

ENO
E3

وتعنــى النتيجة الســابقة أنه عندما يتم دمج رقم الموظــف (E3)، الذى يمثل نتيجة عملية القسمة، مع **كافة** سجلات المشاريع فإن ناتج عملية الدمج هذه ستكون سجلات من ضمن ســجلات جدول الموظفين مما يعنى أن الموظف ذا الرقم (E3) يعمل فى كافة المشاريع.

ومن الأمثلة الأخرى، لنفترض أننا نرغب فى معرفة أســماء أعضاء هيئة التدريس المؤهلين لتدريس كافة المواد الدراســية المؤهل لتدريسها عضو هيئة التدريس «محمود الســالم» (Mahmood Alsalem) (بما فيهم الموظف «محمود الســالم» نفسه). فى هذه

الحالة يتم أولاً الحصول على كافة بيانات الموظف «محمود السالم» (باستخدام عملية اختيـــار) ووضعها فى الجـــدول المؤقت (T_1) (بما فيها رقم الموظف الذى هو الوســـيلة الوحيـــدة لإجراء عملية الربط مع جدول المؤهلات التدريســـية). بعد ذلك يتم إجراء عملية ربط تســـاوى مع جدول المؤهلات التدريســـية متبوعة بعملية إسقاط على حقل «رقم المادة الدراسية» وذلك للحصول على أرقام كافة المواد الدراسية المؤهل لتدريسها عضو هيئة التدريس «محمود السالم»، كما يلى:

$$T_1 \longleftarrow \sigma_{FName = \text{"Mahmood" AND } LName = \text{"Alsalem"}} (FACULTY_T)$$
$$T_2 \longleftarrow \pi_{Course_ID} (QUALIFICATION_T \bowtie T_1)$$

وتكون نتيجة العمليتين السابقتين كما يلى:

T_1

```
FACULTY_ FNAME        LNAME        PHONE_NO     SALARY DOB       DEPART
-------- ------------ ------------ -------- ---------- --------- ------
710      Mahmood      Alsalem      456-3323      31900 19-FEB-73 PHYS
```

T_2

```
COURSE_
-------
PHYS101
```

وللتعرف على المؤهلات التدريسية لكافة أعضاء هيئة التدريس، تتم عملية إسقاط علـــى حقلى «رقـــم عضو هيئة التدريس» و«رقم المادة الدراســـية» فى جدول المؤهلات التدريسية، كما يلى:

$$T_3 \longleftarrow \pi_{Faculty_ID,\ Course_ID} (QUALIFICATION_T)$$

وتكون نتيجة العملية السابقة كما يلى:

T_3

FACULTY_	COURSE_
400	CHEM101
420	CHEM102
310	CS101
320	CS102
320	CS103
330	CS104
340	CS105
800	EE101
810	EE102
850	EE103
810	EE104
500	ENGL101
540	ENGL102
560	ENGL103
200	MATH101
200	MATH102
220	MATH103
220	MATH104
220	MATH106
200	MATH107
710	PHYS101
770	PHYS101
730	PHYS102
600	STAT101
660	STAT101
640	STAT102

بعد ذلك يتم إجراء عملية قسمة بين الجدول المؤقت (T_3) والجدول المؤقت (T_2)؛ للحصول على أرقام أعضاء هيئة التدريس المؤهلين لتدريس نفس المواد الدراسية المؤهل لتدريسها عضو هيئة التدريس «محمود السالم». وللتعرف على أسماء أعضاء هيئة التدريس هؤلاء يتم إجراء عملية ربط طبيعى بين ناتج عملية القسمة (وهو أرقام أعضاء هيئة التدريس المؤهلين لتدريس نفس المواد التى يقوم بتدريسها عضو هيئة التدريس «محمود السالم») بجدول أعضاء هيئة التدريس. بعد ذلك تتم عملية إسقاط على حقلى الاسم الأول واسم العائلة للحصول على جدول النتيجة (Result)، كما يلى:

$$T_4 \longleftarrow T_3 \div T_2$$
$$\text{Result} \longleftarrow \pi_{\text{FName, LName}} (T_4 \bowtie \text{FACULTY_T})$$

وتكون نتيجة العملية الأولى كما يلى:

T_4

```
FACULTY_
--------
710
770
```

أما النتيجة النهائية لما هو مطلوب فتكون كما يلى:

Result

```
FNAME        LNAME
------------ ------------
Mahmood      Alsalem
Sultan       Aljasir
```

٤-٣ الحساب العلاقى (Relational Calculus):

يستعرض هذا الجزء الحساب العلاقى الذى يعد اللغة الرسمية الثانية للنموذج العلاقى. وعلى الرغم من وجود نوعين من الحساب العلاقى وهما: الحساب العلاقى المتعلق بالسجلات (Tuple Relational Calculus)، والحساب العلاقى المتعلق بالمجال (Domain Relational Calculus)، إلا أننا سنستعرض النوع الأول فقط من الحساب العلاقى، وذلك لأن اللغة السائدة فى التعامل مع بيانات قواعد البيانات العلاقية، وهى لغة الاستفسار البنائية (SQL)، مبنية على الحساب العلاقى المتعلق بالسجلات. وقد تم تطوير كلا النوعين من الحساب العلاقى، بشكل متزامن تقريباً، فى مركزين من مراكز أبحاث شركة آى بى أم. أما بالنسبة للحساب العلاقى المتعلق بالمجال فقد تم تطويره إلى لغة أخرى تتعامل مع قواعد البيانات العلاقية وهى لغة «الاستفسار بالمثال» ((Query-By-Example (QBE)) التى من أمثلتها تلك المستخدمة فى نظام إدارة قاعدة بيانات أكسس المصنعة من قبل شركة ميكروسوفت للحاسبات الشخصية.

إن الحساب العلاقى، وعلى النقيض من الجبر العلاقى الذى سبق شرحه فى الجزء السابق، لغة وصفية (Declarative Language) تستخدم لتحديد استفسار ما دون توصيف للطريقة الواجب اتباعها للحصول على نتيجة عملية الاستفسار. ويعنى

هذا أن أى تعبير حسابى علاقى يحدد ماهية البيانات الواجب استرجاعها عوضاً عن كيفية الوصول إلى البيانات. لذا فإن الحساب العلاقى يعد لغة لا إجرائية (Nonprocedural Language). وعلى خلاف ذلك فإن الجبر العلاقى يعد لغة إجرائية (Procedural Language)؛ لأنه يجب علينا كتابة سلسلة من العمليات التى يلزم تنفيذها للوصول إلى البيانات المطلوبة وأن سلسلة العمليات هذه تحدد ترتيباً للعمليات الموجودة فى السلسلة. أما فى الحساب العلاقى فإنه يمكن صياغة الاستفسار بأكثر من طريقة، ومع ذلك فإن صياغة الاستفسار لا تؤثر على كيفية الوصول للبيانات المطلوبة. ولذلك فإن اللغة المستخدمة للتعامل مع قواعد البيانات، وهى لغة الاستفسار البنائية (SQL)، مبنية على أساس هذا النوع من اللغات الرسمية للنموذج العلاقى؛ لكونه يعفى المستفيدين من الخوض فى تفاصيل كيفية الوصول للبيانات التى يرغبون فى الحصول عليها كما يترك المجال مفتوحاً للشركات المطورة لنظم إدارة قواعد البيانات لتطوير طرق مختلفة لتنفيذ استفسارات المستفيدين بشكل فعال. وتعد سرعة الاستجابة للاستفسارات المختلفة من قبل نظم إدارة قواعد البيانات أحد المعايير المهمة التى تميز بين نظام وآخر، وذلك نتيجة للاختلافات فى فاعلية الطرق المستخدمة من قبل هذه النظم فى كيفية الوصول إلى البيانات المطلوبة من قبل استفسارات المستفيدين.

٤-٣-١ متغيرات السجلات (Tuple Variables):

يعتمد الحساب العلاقى على ما يعرف بمتغيرات السجلات، بحيث إن كل متغير (Variable) يأخذ قيم سجلات جدول واحد لسجل تلو الآخر، بمعنى أن أى متغير يتم تعريفه يجوب كل سجلات جدول واحد من جداول قاعدة البيانات. فعلى سبيل المثال، الصيغة التالية تمثل استفساراً مكتوباً بالحساب العلاقى:

$$\{t \mid t \in \text{FACULTY_T}\}$$

وتعنى الصيغة أعلاه أن المتغير (t) قد تم تعريفه، بحيث يجوب جدول أعضاء هيئة التدريس (وهو المقصود فى الجانب الأيمن من الصيغة) وستكون نتيجة الاستفسار (وهى الجانب الأيسر من الصيغة) كافة سجلات جدول أعضاء هيئة التدريس. وتقرأ الصيغة السابقة كما يلى: ما هى السجلات (t) بحيث إن التى يرمز لها بالرمز (|)، هذه السجلات التى سيجوبها المتغير (t) تنتمى لجدول أعضاء هيئة التدريس. ويمكن أن توضع بعض الشروط على السجلات الواجب استرجاعها من جدول أعضاء هيئة

التدريس. فعلى سبيل المثال، للحصول على بيانات أعضاء هيئة التدريس الذين يعملون فى قسم الحاسب الآلى فقط، يمكن صياغة الاستفسار كما يلى:

$$\{t \mid t \in \text{FACULTY_T} \wedge t\,[\text{Department_ID}] = \text{"CS"}\}$$

لقد تم تعريف المتغير (t) فى الصيغة السابقة، بحيث يجوب كافة سجلات جدول أعضاء هيئة التدريس، وعندما يكون حقل «رقم القسم الدراسى» للسجل الذى قد أخذ قيمته المتغير مساوياً لقسم الحاسب الآلى، يكون هذا السجل من ضمن سجلات نتيجة الاستفسار. ويمكن النظر للمتغير على أنه مؤشر يشير لسجلات الجدول الواحد تلو الآخر، وعندما تتحقق الشروط الواردة فى صيغة الاستفسار، وهى أن يكون القسم الذى يتبعه عضو هيئة التدريس هو قسم الحاسب الآلى فى مثالنا، يكون السجل من ضمن سجلات نتيجة الاستفسار. إن الصيغة السابقة للاستفسار تسترجع كافة حقول السجلات التى ينطبق عليها شرط الاسترجاع، إلا أنه بالإمكان استرجاع بعض حقول السجلات عوضاً عن جميع الحقول. فعلى سبيل المثال، لمعرفة الاسم الأول واسم العائلة لكل عضو هيئة تدريس يعمل فى قسم الحاسب الآلى، يمكن كتابة الاستفسار وفق الصيغة التالية:

$$\{t\,[\text{FName}], t\,[\text{LName}] \mid t \in \text{FACULTY_T} \wedge t\,[\text{Department_ID}] = \text{"CS"}\}$$

وبشكل عام نحتاج إلى تحديد المعلومات التالية فى الحساب العلاقى:

١- لكل متغير نحتاج إلى تحديد الجدول (R) الذى سيجوبه هذا المتغير وفق الصيغة: $t \in R$.

٢- تحديد شرط الاسترجاع، فعندما يجوب المتغير سجلات الجدول الذى تم تعريف المتغير عليه، يتم فحص شرط الاسترجاع، فإذا كانت النتيجة صحيحة (True)، بمعنى انطباق الشرط على السجل قيد الفحص، يتم إدراج السجل ضمن نتيجة الاستفسار.

٣- تحديد الحقول الواجب إظهارها ضمن نتيجة الاستفسار، بحيث يتم إدراج هذه الحقول ضمن نتيجة الاستفسار لكل سجل ينطبق عليه شرط الاستفسار.

فعلى سبيل المثال، للحصول على رقم هاتف وتاريخ ميلاد كل عضو هيئة تدريس يعمل فى الجامعة الأهلية واسم عائلته هو «الصالح» (Alsaleh) وأن القسم الذى يعمل فيه ليس قسم الحاسب الآلى؛ يمكن كتابة الاستفسار كما يلى:

$$\{t\,[\text{Phone_No}],\ t\,[\text{DOB}] \mid t \in \text{FACULTY_T} \wedge t\,[\text{LName}] = \text{“Alsaleh”} \wedge t\,[\text{Department_ID}] \neq \text{“CS”}\}$$

وتعنى الصيغة السابقة أننا نقوم بتحديد الحقول التى ستنتج عن عملية الاستفسار وهى «رقم الهاتف»، و«تاريخ الميلاد» لكل سجل (t) يتم اختياره. بعد ذلك نقوم بتحديد الشروط الواجب أن تتحقق فى السجل (بعد الرمز (|))، وهى فى مثالنا أن يكون اسم العائلة «الصالح» (Alsaleh) وأن القسم الذى يعمل فيه ليس قسم الحاسب الآلى.

٤-٣-٢ التعابير والتراكيب فى الحساب العلاقى (Expression and Formulae in Relational Calculus):

إن الشكل العام للتعابير فى الحساب العلاقى هو كما يلى:

$$\{t \mid F(t)\}$$

بحيث إن الجانب الأيسر للرمز (|) يمثل متغيراً ، والجانب الأيمن من الرمز عبارة عن تركيبة أو صيغة (Formula) يمكن أن تكون نتيجتها متحققة (True) أو غير متحققة (False). كما يمكن أن يقيد المتغير بحقل أو أكثر مثل: $t[A_1]$ و $t[A_2]$ بحيث إن A_1 و A_2 تمثل حقلين من حقول السجلات فى الجدول الذى يجوبه المتغير. وعندما تتحقق التركيبة أو الصيغة على السجل الذى قد أخذ المتغير قيمته، يكون الحقلان من ضمن نتيجة الاستفسار. وتتكون التركيبة من وحدات أو ذرات (Atoms) وعوامل حسابية ومنطقية. وتكون الوحدات كما يلى:

- **متغيرات:**

- عندما يكون الجدول الذى يجوبه المتغير معروفاً، من الممكن أن يقيد المتغير باسم الجدول، وليكن (R)، كما يلى: R.t أو R(t) أو $t \in R$. (وسنستخدم الصيغة الأخيرة فى هذا الكتاب لتمثيل الوحدات التى تعرف المتغيرات).

- **شروط:**

- عندما يتواجد متغيران، وليكونا s و t، يمكن أن يربط بينهما بأحد عوامل المقارنة الحسابية θ، وهى: $\theta \in \{=, >, \geq, <, \leq, \neq\}$ ، على الشكل التالى: $t[A]\ \theta\ s[B]$.

- عوضـاً عن ربـط متغير ما بمتغير آخر باسـتخدام عوامل المقارنة الحسـابية السـابقة، فإنه يمكن أن يربط المتغير، وليكن t، بقيمة ثابتة (Constant)، ولتكن c، كما يلى: $t[A]\ \theta\ c$.

وباسـتخدام الوحدات حسـب تعريفها أعلاه يمكن أن تؤلف التركيبات (Formulae) بحيث تتكون كل تركيبة مما يلى:

- وحدات.
- عوامل منطقية: $\neg, \wedge, \vee$.
- عامل الوجود (Existential Quantifier): $\exists$.
- عامل الكل (Universal Quantifier): $\forall$.

أما قواعد تكوين الصيغ فهى كما يلى:

- كل وحدة تمثل صيغة.
- إذا كانت F و G صيغتين فإن $(F \wedge G)$ و $(F \vee G)$ و$(\neg F)$ و$(\neg G)$ صيغ أيضاً.
- إذا كانت F صيغة فكذلك (F).
- إذا كانـت F صيغـة وكان t متغيراً فى الصيغة فـإن $\exists t(F)$ صيغة أيضاً تكون نتيجتها صحيحة (True) إذا وُجد سجل واحد على الأقل فى سجلات الجدول الذى تم تعريف المتغير عليه بحيث تنطبق عليه الصيغة F. وخلاف ذلك تكون النتيجة خطأ (False). كما يمكن كتابة الصيغة $\exists t(F)$ على الشكل $\exists tF(t)$.
- إذا كانت F صيغة وكان t متغيراً فى الصيغة فإن $\forall t(F)$ تعد صيغة أيضاً تكون نتيجتهـا صحيحة (True) إذا انطبقت الصيغة F على كافة سـجلات الجدول الـذى تم تعريف المتغير عليه. وخلاف ذلك تكون النتيجة خطأ (False). كما يمكن كتابة الصيغة $\forall t(F)$ على الشكل $\forall tF(t)$.

٤-٣-٢-١ التعابير الآمنة (Safe Expressions):

عندما يسـتخدم عامل الوجود، وعامل الكل، ونفى الشروط فى تعبيرات الحساب العلاقـى فإنه من الضرورى التأكد من أن التعبيرات الحسـابية ذات معنى. والتعبير الآمـن فى الحسـاب العلاقى يضمن أن تكـون نتيجته عدداً محدداً من السـجلات.

وخلاف ذلك فإن التعبير الحسابى يعد غير آمن. فعلى سبيل المثال، يعد التعبير التالى غير آمن لكون نتيجته عدد غير محدد من السجلات:

$$\{t \mid \neg\, t \in R\}$$

وتكون نتيجة التعبير السابق هى كافة السجلات التى من الممكن أن توجد ولكنها ليست من ضمن السجلات الموجودة فعلياً فى الجدول (R). وهذه السجلات بالطبع ذات عدد غير محدود. ويمكن تعريف التعبير الآمن بشكل أكثر تحديداً، كما يلى:

> يعد التعبير الحسابى آمناً إذا كان بالإمكان حساب نتيجته باستخدام قيم ثابتة فقط من ضمن القيم الموجودة فى قاعدة البيانات أو من ضمن التعبير الحسابى نفسه.

ولكون قاعدة البيانات تحتوى على عدد محدد من القيم، فإن القيم الثابتة المخزنة فيها محدودة. وكذلك هو الحال بالنسبة لعدد القيم الثابتة التى من الممكن أن يحتويها التعبير الحسابى التى لا بد أن تكون محدودة أيضاً. ونتيجة لذلك فإن التعبير الحسابى سيكون آمناً من حيث إن نتيجته ستحتوى على عدد محدد من القيم إذا احتوى على قيم ثابتة من قيم قاعدة البيانات أو قيم ثابتة ضمنه.

ولنفترض وجود الجداول الأربعة التالية ضمن قاعدة البيانات التى تمثل جدول الموظفين، وجدول المشاريع، وجدول المخصصات المالية، وجدول الأعمال التى يقوم بها كل موظف ضمن كل مشروع يشارك فيه. ولنفترض كذلك أننا نرغب فى إجراء بعض الاستفسارات.

Employee (Eno, Ename, Title, City)

Project (Pno, Pname, Budget, City)

Payment (Title, Salary)

Job (Eno, Pno, Responsibility, Duration)

الاستفسار الأول: ما أسماء كافة الموظفين؟

الحل:

$\{t\,[Ename] \mid t \in Employee\}$

اقترن المتغير (t) فى الجانب الأيســر من التعبير الحسابى، فى الحل أعلاه، بحقل «اســم الموظــف» (Ename) مما يعنى أن ســجلات النتيجة النهائية للتعبير الحســابى ستتضمن حقل «اسم الموظف» فقط، وذلك لكل سجل تنطبق عليه الصيغة المعرفة فى الجانب الأيمن من التعبير الحســابى. أما الصيغة التى يجب أن تتحقق على الســجل حتى يكون من ضمن سجلات النتيجة النهائية للتعبير الحسابى فلم تتضمن أية شروط. لذلك فإنه عندما يجوب المتغير (t) ســجلات جدول الموظفين ($t \in$ Employee)، الواحد تلو الآخر، ســيكون كل ســجل من ســجلاته مؤهلاً لأن يكون من ضمن نتيجة التعبير الحسابى. وبناءً على ذلك ســتحتوى النتيجة النهائية للتعبير الحسابى على سجلات تتضمن كافة أسماء الموظفين.

الاستفسار الثانى: ما أسماء المشاريع وميزانياتها؟

الحل:

$\{t\,[\text{Pname}],\, t\,[\text{Budget}] \mid t \in \text{Project}]\}$

اقترن المتغير (t) فى الجانب الأيسر من التعبير الحسابى، فى الحل أعلاه، بحقل كل من «اســم المشروع» (Pname) وحقل «ميزانية المشروع» (Budget)؛ مما يعنى أن سجلات النتيجة النهائية للتعبير الحســابى ســتتضمن حقل «اسم المشــروع» وحقل «ميزانية المشــروع» فقط، وذلك لكل ســجل تنطبق عليه الصيغة المعرفة فى الجانب الأيمن من التعبير الحسابى. أما الصيغة التى يجب أن تتحقق على السجل حتى يكون من ضمن سجلات النتيجة النهائية للتعبير الحسابى فلم تتضمن أية شروط. لذلك فإنه عندما يجوب المتغير (t) ســجلات جدول المشاريع ($t \in$ Project)، الواحد تلو الآخر، سيكون كل سجل من سجلاته مؤهلاً لأن يكون من ضمن نتيجة التعبير الحسابي. وبناءً على ذلك ستتضمن النتيجة النهائية للتعبير الحسابى سجلاتٍ تحتوى على كافة أسماء المشاريع وميزانياتها.

الاستفســار الثالث: ما مســميات الوظائف التى تم تعيين موظف واحد على الأقل فى كل واحدة منها؟

الحل:

$\{t\,[\text{Title}] \mid t \in \text{Employee})\}$

يحتوى جدول الموظفين على سجلات لكافة الموظفين متضمناً ذلك مسميات الوظائف التى تم تعيينهم عليها. ويعنى هذا أن كل مسمى وظيفى مدرج فى جدول الموظفين يعنى ضمنياً وجود موظف واحد على الأقل معين على مسمى هذه الوظيفة. ولحل الاستفسار، فى هذه الحالة، يكتفى بمعرفة مسميات الوظائف المدرجة فى جدول الموظفين. ولمعرفة مسميات الوظائف هذه، اقترن المتغير (t) فى الجانب الأيسر من التعبير الحسابى بحقل «مسمى الوظيفة» (Title)، فى الحل أعلاه، مما يعنى أن سجلات النتيجة النهائية للتعبير الحسابى ستتضمن حقل «مسمى الوظيفة» فقط، وذلك لكل سجل تنطبق عليه الصيغة المعرفة فى الجانب الأيمن من التعبير الحسابي. أما الصيغة التى يجب أن تتحقق على السجل حتى يكون من ضمن سجلات النتيجة النهائية للتعبير الحسابى فلم تتضمن أى شروط. لذلك فإنه عندما يجوب المتغير (t) سجلات جدول الموظفين ($t \in$ Employee)، الواحد تلو الآخر، سيكون كل سجل من سجلاته مؤهلاً لأن يكون من ضمن نتيجة التعبير الحسابى. وبناءً على ذلك ستتضمن النتيجة النهائية للتعبير الحسابى سجلات تحتوى على كافة مسميات الوظائف للموظفين ذوى السجلات فى جدول الموظفين. وهذا يعنى أن النتيجة ستحتوى على مسميات كافة الوظائف التى تم تعيين موظف واحد على الأقل على كل منها.

الاستفسار الرابع: ما بيانات الموظفين الذين يقطنون فى مدينة الدمام؟

الحل:

$$\{t \mid t \in \text{Employee} \wedge t[\text{City}] = \text{“Dammam”}\}$$

لم يقترن المتغير (t) فى الجانب الأيسر من التعبير الحسابى، فى الحل أعلاه، بأى حقل من حقول سجلات الجدول الذى يجوبه المتغير. لذلك فإن النتيجة النهائية للتعبير الحسابى ستتضمن كافة حقول سجلات الجدول التى تنطبق عليها الصيغة المعرفة فى الجانب الأيمن من التعبير الحسابى. أما الصيغة التى يجب أن تتحقق على السجل حتى يكون من ضمن سجلات النتيجة النهائية للتعبير الحسابى فتضمنت شرط كون قيمة حقل «المدينة» (City) مساوية لمدينة «الدمام» (Dammam). لذلك فإنه عندما يجوب المتغير (t) سجلات جدول الموظفين ($t \in$ Employee)، الواحد تلو الآخر، سيكون كل سجل قيمة حقل المدينة فيه مساوية لمدينة الدمام مؤهلاً لأن يكون من ضمن نتيجة التعبير الحسابى. وبناءً على ذلك ستتضمن النتيجة النهائية للتعبير الحسابى كافة بيانات الموظفين الذين يقطنون فى مدينة الدمام.

الاستفسار الخامس: ما المدن التى يقطنها موظفون ويتوافر فيها مشاريع؟

الحل:

$$\{t\,[\text{City}] \mid t \in \text{Employee} \land \exists s\,(s \in \text{Project} \land t\,[\text{City}] = s\,[\text{City}])\}$$

أو

$$\{t\,[\text{City}] \mid t \in \text{Project} \land \exists s\,(s \in \text{Employee} \land t\,[\text{City}] = s\,[\text{City}])\}$$

لحل هذا الاستفسار نحتاج إلى تعريف متغيرين: الأول منهما يجوب جدول الموظفين، والثانى يجوب جدول المشاريع. وعندما يكون حقل المدينة فى السجل الذى أخذ قيمته المتغير الذى يجوب جدول الموظفين مساوياً لحقل المدينة فى أحد سجلات المتغير الذى يجوب جدول المشاريع، فإن هذا يعنى وجود موظف ومشروع فى نفس المدينة وتكون المدينة من ضمن نتيجة التعبير الحسابى.

اقترن المتغير (t) فى الجانب الأيسر من التعبير الحسابى، فى الحل الأول أعلاه، بحقل «المدينة» (City)، مما يعنى أن سجلات النتيجة النهائية للتعبير الحسابى ستتضمن حقل «المدينة» فقط، وذلك لكل سجل تنطبق عليه الصيغة المعرفة فى الجانب الأيمن من التعبير الحسابى. أما الصيغة التى يجب أن تتحقق على السجل حتى يكون من ضمن سجلات النتيجة النهائية للتعبير الحسابى فشرحها كما يلى:

- يجوب المتغير (t) جدول الموظفين السجل تلو الآخر. ويعنى هذا أن حقل «المدينة» فى النتيجة النهائية هو حقل «المدينة» التابع لجدول الموظفين لكون الحقل مرتبطاً بالمتغير (t) الذى تم تعريفه بحيث يجوب سجلات جدول الموظفين.
- لكل سجل يأخذ قيمته المتغير (t) يجب أن يتحقق الشرط التالى:

* يجوب المتغير (s) جدول المشاريع، وعندما يوجد (∃) سجل فى جدول المشاريع تكون المدينة الموجودة فيه مساوية لمدينة الموظف فى السجل الذى توقف عنده المتغير (t)، يكون سجل الموظف من ضمن سجلات جدول النتيجة النهائية التى يؤخذ منها قيمة حقل المدينة فقط.

- تستمر العملية حتى ينتهى المتغير (t) من المرور بكافة سجلات الموظفين.

الحل الثانى للاستفسار مماثل للحل الأول ومكافئ لنتيجته، إلا أنه يلاحظ فى الحل الثانى عكس ترتيب الجداول التى يجوبها المتغيران، فالمتغير (t) يجوب جدول

المشـــاريع عوضاً عن جدول الموظفين، والمتغير (s) يجوب جـــدول الموظفين عوضاً عن جدول المشاريع. أما حقل المدينة، الذى يمثل نتيجة التعبير الحسابى فهو حقل جدول المشاريع عوضاً عن حقل جدول الموظفين.

الاستفسـار السـادس: مـــا المـــدن التـــى يوجد فيهـــا مشـــاريع ولا يقطنها أى من الموظفين؟

الحل:

$$\{t\,[\text{City}] \mid t \in \text{Project} \wedge \neg \exists s\,(s \in \text{Employee} \wedge t\,[\text{City}] = s\,[\text{City}])\}$$

لحـــل هذا الاستفســـار نحتاج إلى تعريـــف متغيرين: الأول منهمـــا يجوب جدول المشـــاريع، والثانى يجوب جـــدول الموظفين. وعندما **لا يوجـد** لقيمة حقل المدينة فى الســـجل الذى أخذ قيمته المتغير الذى يجوب جدول المشاريع ما يساويها من قيمة فى حقل المدينة فى كافة سجلات المتغير الذى يجوب جدول الموظفين فإن هذا يعنى وجود مشروع فى المدينة مع عدم وجود ما يقطنها من موظفين.

اقترن المتغير (t) فى الجانب الأيســـر من التعبير الحسابى، فى الحل أعلاه، بحقل «المدينة» (City)، مما يعنى أن ســـجلات النتيجة النهائية للتعبير الحســـابى ستتضمن حقل «المدينة» فقط، وذلك لكل ســـجل تنطبق عليه الصيغة المعرفة فى الجانب الأيمن من التعبير الحســـابى. أما الصيغة التى يجب أن تتحقق على الســـجل حتى يكون من ضمن سجلات النتيجة النهائية للتعبير الحسابى فشرحها كما يلى:

- يجوب المتغير (t) جدول المشاريع السجل تلو الآخر. ويعنى هذا أن حقل «المدينة» فى النتيجة النهائية هو حقل «المدينة» التابع لجدول المشاريع لكون الحقل مرتبطاً بالمتغير (t) الذى تم تعريفه بحيث يجوب سجلات جدول المشاريع.

- لكل ســـجل يأخذ قيمته المتغير (t) يجب أن **لا يتحقق** الشرط التالى (بمعنى أن السجل لن يكون من ضمن النتيجة النهائية للتعبير الحسابى):

* يجـــوب المتغير (s) جـــدول الموظفين، وعندما يوجد (∃) ســـجل فى جدول الموظفين تكون المدينة الموجودة فيه مســـاوية لمدينة المشـــروع فى السجل الذى توقف عنده المتغير (t)، يكون سجل المشروع من ضمن سجلات جدول النتيجة النهائية التى يؤخذ منها قيمة حقل المدينة فقط.

- تستمر العملية حتى ينتهى المتغير (t) من المرور بكافة سجلات الموظفين.

وتتم عملية اســتثناء ســجلٍ ما من الدخول فى النتيجة النهائية للتعبير الحســابى باســتخدام العامل المنطقى (¬)؛ مما يعنى أن عامل النفى إذا سبق عامل الوجود تكون نتيجته «عدم الوجود».

ويلاحــظ فى هذا المثــال عدم وجود حل ثانٍ مماثل لحل المثــال الخامس؛ لأننا لو عكســنا ترتيب الجــداول المرتبطة بالمتغيرات فســتكون النتيجة مكافئة للاستفســار التالى:

الاستفسار: ما المدن التى يقطنها موظفون ولا يوجد فيها مشاريع؟

ونتيجــة الاستفســار الســابق مختلفة، بكل تأكيــد، عن المطلوب فى الاستفســار الأساسى.

الاستفسار السابع: ما أسماء المشاريع التى تزيد ميزانياتها على (٢٥٠،٠٠٠)؟

الحل:

$$\{t\,[\text{Pname}] \mid t \in \text{Project} \wedge t\,[\text{Budget}] > 250000\}$$

اقترن المتغير (t) فى الجانب الأيســر من التعبير الحسابى، فى الحل أعلاه، بحقل «اســم المشــروع» (Pname) مما يعنى أن ســجلات النتيجة النهائية للتعبير الحســابى ستتضمن حقل «اسم المشروع» فقط، وذلك لكل سجل تنطبق عليه الصيغة المعرفة فى الجانب الأيمن من التعبير الحســابي. أما الصيغة التى يجب أن تتحقق على الســجل حتى يكون من ضمن ســجلات النتيجة النهائية للتعبير الحسابى فتضمنت شرط كون قيمة حقل «ميزانية المشروع» (Budget) أكبر من (٢٥٠،٠٠٠). لذلك فإنه عندما يجوب المتغير (t) ســجلات جدول المشــاريع ($t \in$ Project)، الواحد تلو الآخر، سيكون كل سجل قيمــة حقــل ميزانيته أكبر من (٢٥٠،٠٠٠) مؤهل لأن يكون مــن ضمن نتيجة التعبير الحســابى. وبناءً على ذلك ســتتضمن النتيجة النهائية للتعبير الحسابى أسماء كافة المشاريع التى تزيد ميزانياتها على (٢٥٠،٠٠٠).

الاستفسار الثامن: ما أسماء وميزانيات المشاريع التى يعمل فيها الموظف رقم (E1)؟

الحل:

$$\{t\,[\text{Pname}],\ t\,[\text{Budget}] \mid t \in \text{Project} \wedge \exists s\,(s \in \text{Job} \wedge t\,[\text{Pno}]=s\,[\text{Pno}] \wedge s\,[\text{Eno}] = \text{"E1"})\}$$

اقترن المتغير (t) فى الجانب الأيسر من التعبير الحسابى، فى الحل أعلاه، بحقل كل من «اســم المشروع» (Pname) وحقل «ميزانية المشروع» (Budget)، مما يعنى أن سجلات النتيجة النهائية للتعبير الحســابى ســتتضمن حقل «اسم المشــروع» وحقل «ميزانية المشــروع» فقط، وذلك لكل ســجل تنطبق عليه الصيغة المعرفة فى الجانب الأيمن من التعبير الحسابى. أما الصيغة التى يجب أن تتحقق على السجل حتى يكون من ضمن سجلات النتيجة النهائية للتعبير الحسابى فشرحها كما يلى:

- يجوب المتغير (t) جدول المشاريع السجل تلو الآخر. ويعنى هذا أن كل من حقل «اســم المشروع» (Pname) وحقل «ميزانية المشروع» (Budget) فى النتيجة النهائية للتعبير الحسابى هما من حقول جدول المشاريع؛ وذلك لكونهما مرتبطين بالمتغير (t) الذى تم تعريفه بحيث يجوب سجلات جدول المشاريع.

- لكل سجل يأخذ قيمته المتغير (t) يجب أن يتحقق الشرط التالى:

* يجوب المتغير (s) جدول «الأعمال» (Job)، وعندما يوجد (∃) سجل فى جدول الأعمال تكون قيمة حقل «رقم المشروع» (Pno) فيه مساوية لقيمة حقل «رقم المشــروع» (Pno) فى السجل الذى توقف عنده المتغير (t)، وتكون قيمة حقل «رقم الموظف» (Eno) فى السجل الذى أخذ قيمته المتغير (s) هى "E1"، يكون سجل المشروع من ضمن ســجلات جدول النتيجة النهائية التى يؤخذ منها قيمة حقل «اسم المشروع» وحقل «ميزانية المشروع».

- تستمر العملية حتى ينتهى المتغير (t) من المرور بكافة سجلات المشاريع.

٤-٤ أمثلة على استخدام الجبر العلاقى والحساب العلاقى:

يتوافر لإحدى مؤسسات تأجير العقارات عدد من المكاتب فى مدن مختلفة، وعدد من العاملين فى المؤسســة. كما تحتفظ المؤسسة ببيانات عن عملائها الذين يتقدمون لها بطلبات لاســتئجار العقارات التى تشــرف عليها بالإضافة لبيانات العقارات التى تشــرف عليها وبيانــات مالكى هذه العقارات. كما أن المؤسســة تحتفظ ببيانات عن المواعيد التى تم تحديدها للعملاء المختلفين لمعاينة العقارات التى تشــرف عليها قبل قيامهم باستئجار ما يتناسب من العقارات التى تشرف عليها المؤسسة مع احتياجاتهم. وتحتفظ المؤسسة بهذه البيانات ضمن قاعدة بيانات علاقية تتكون من الجداول التالية :(Connolly and Begg, 2000)

١- جدول مكاتب المؤسسة:

Branch (BranchNo: integer, Street: string, City: string, Postcode: string)

٢- جدول العاملين فى المؤسسة:

Staff (StaffNo: integer, FName: string, LName: string, Position: string, Sex: string, DOB: date, Salary: integer, BranchNo: integer)

٣- جدول العقارات التى تشرف عليها المؤسسة:

PropertyForRent (PropertyNo: integer, Street: string, City: string, Postcode: integer, Type: string, NumberOfRooms, integer, Rent: integer, OwnerNo: integer, StaffNo: integer, BranchNo: integer)

٤- جدول العملاء (أو المستأجرين):

Client (ClientNo: integer, FName: string, LName: string, TelNo: string, PropertyPreference: string, MaxRent: integer)

٥- جدول ملاك العقارات:

Owner (OwnerNo: integer, FName: string, LName: string, Address: string, TelNo: string)

٦- جدول مواعيد معاينة العقارات من قبل العملاء (تدخل المواعيد بعد الانتهاء من المعاينة):

ViewingSchedule (ClientNo: integer, PropertyNo: integer, ViewDate: date, Comments: string)

وبناءً على الجداول السابقة لقاعدة بيانات المؤسسة، المطلوب هو الإجابة عن الاستفسارات التالية باستخدام (١) الجبر العلاقى و(٢) الحساب العلاقى.

الاستفسار الأول: ما الأسماء الأولى وأسماء عائلات العاملين الذين يشغلون مناصب إشرافية فى المؤسسة ويتقاضون رواتب تزيد على (٥٠,٠٠٠)؟

الحل:

1- $\pi_{FName,\ LName}\ (\sigma_{Position\ =\ \text{“Manager”}\ \wedge\ Salary\ >\ 50000}\ (Staff))$

2- $\{t\,[FName],\ t\,[LName]\ |\ t \in Staff\ \wedge\ t\,[Position] = \text{“Manager”}\ \wedge\ t\,[Salary] > 50000\}$

الاستفسار الثانى: ما الأسماء الأولى وأسماء عائلات العاملين الذين يشرفون على عقارات معروضة للإيجار فى مدينة الرياض؟

الحل:

1- $\pi_{FName, LName} (\sigma_{Staff.StaffNo = PropertyForRent.StaffNo} (Staff \times \sigma_{City = \text{"Riyadh"}} (PropertyForRent)))$

ملاحظة: يمكن استخدام إعادة تسمية حقل «رقم الموظف» فى أحد الجدولين عوضاً عن استخدام اسم الجدول متبوعاً باسم الحقل لحل هذا المثال، كما يلى:

1- $\pi_{FName, LName} (\sigma_{StaffNo = Staff_ID} (Staff \times \rho_{StaffNo = Staff_ID} (\sigma_{City = \text{"Riyadh"}} (PropertyForRent))))$

2- $\{t[FName], t[LName] \mid t \in Staff \wedge \exists s (s \in PropertyForRent \wedge s[StaffNo] = t[StaffNo] \wedge s[City] = \text{"Riyadh"})\}$

الاستفسار الثالث: ما الأسماء الأولى وأسماء عائلات العاملين فى المؤسسة الذين لا يشرفون حالياً على أية عقارات معروضة للإيجار؟

الحل:

1- $\pi_{FName, LName} (Staff - (Staff \bowtie \pi_{StaffNo}(PropertyForRent)))$

2- $\{t[FName], t[LName] \mid t \in Staff \wedge \neg \exists s (s \in PropertyForRent \wedge s[StaffNo] = t[StaffNo])\}$

الاستفسار الرابع: ما الأسماء الأولى وأسماء عائلات العملاء الذين قاموا بمعاينة عقارات فى مدينة الرياض؟

الحل:

1- $\pi_{FName, LName} (Client \bowtie (ViewingSchedule \bowtie \sigma_{City = \text{"Riyadh"}} (PropertyForRent)))$

2- $\{t[FName], t[LName] \mid t \in Client \wedge \exists s \exists u (s \in ViewingSchedule\ u \in PropertyForRent \wedge s[ClientNo] = t[ClientNo] \wedge s[PropertyNo] = u[PropertyNo] \wedge u[City] = \text{"Riyadh"})\}$

الاستفسار الخامس: ما المدن التى يتوافر فيها مكتب للمؤسسة وعقار واحد على الأقل معروضاً للإيجار فيها؟

الحل:

1- $\pi_{City}(Branch) \cap \pi_{City}(PropertyForRent)$

2- $\{t[City] \mid t \in Branch \wedge \exists s\,(s \in PropertyForRent \wedge s[City] = t[City])\}$

الفصل الخامس

التصميم المنطقى لنظم قواعد البيانات العلاقية

تتكون مرحلة التصميم المنطقى لقواعد البيانات من خطوتين رئيسيتين. فى الخطوة الأولى يتم تحويل النموذج المفاهيمى إلى نموذج قاعدة البيانات المستخدمة. ولأن هذا الكتاب يركز على قواعد البيانات العلاقية، فإن هذه الخطوة تعنى تحويل النموذج المفاهيمى إلى النموذج العلاقى. أما فى الخطوة الثانية فيتم تحسين تصميم قاعدة البيانات الناتجة من عملية التحويل بحيث تحتوى على أقل قدر ممكن من البيانات المتكررة حتى يتم تجنب الأخطاء التى قد تنتج عن عمليات التعديل على محتويات قاعدة البيانات. وتدعى هذه الخطوة بعملية «التطبيع» (Normalization).

ويركز هذا الفصل من الكتاب على الخطوة الأولى من عملية التصميم المنطقى لقواعد البيانات، بحيث يتم تصميم هياكل قاعدة البيانات العلاقية بناءً على تصميمها المسبق باستخدام النموذج المفاهيمى. ويطلق على هذه الخطوة فى بعض الأحيان عملية «التحويل بين النماذج». ولا تقتصر هذه الخطوة على قواعد البيانات العلاقية، ولكن يمكن أن تطبق على أى نماذج أخرى لقواعد البيانات التمثلية مثل الشبكية، والهرمية، والشيئية. فهذه الخطوة من مرحلة التصميم المنطقى ما هى إلا عملية تصميم لهياكل قاعدة البيانات وفق النموذج التمثيلى المستخدم. ولأن النموذج العلاقى هو أحد المحاور الرئيسية لهذا الكتاب، فإن هذا الفصل يركز على عملية التصميم المنطقى لقواعد البيانات العلاقية. ولكون عملية تصميم نظم قواعد البيانات تأتى عادة بشكل متسلسل، فإن هذا الجزء يركز على عملية تحويل النموذج المفاهيمى، القريب من مستوى إدراك وفهم المستفيدين لقاعدة البيانات التى تمثل البيانات التى يتعاملون معها والقيود المفروضة عليها، إلى النموذج العلاقى، الذى يتم التعامل معه من قبل المتخصصين فى نظم قواعد البيانات والمستفيدين ذوى الخبرة فى مجال الحاسب الآلى.

ويعتمد العديد من أدوات هندسة البرمجيات (Computer-Aided Software Engineering (CASE) Tools) على نموذج كينونة - علاقة، أو نماذج شبيهة، فى عملية التصميم

المفاهيمـــى لقواعد البيانات. وتمكن هـــذه الأدوات مصممى نظم قواعد البيانات من تصميم قواعد البيانات بشـــكل تفاعلى (Interactive) على هيئة رسومات نموذج كينونة - علاقة، التى ســـبق شـــرحها فى الفصلين الثانى والثالث. كما تمكن هذه الأدوات، وبشـــكل آلى، من تحويل النموذج المفاهيمى الذى تم تصميمه إلى هياكل قاعدة بيانات علاقية مســـتخدمة لغة تعريف البيانـــات (Data Definition Language)، التى تعد جزءاً مـــن لغة الاستفســـار البنائية (SQL)، الخاصة بقاعدة البيانات العلاقية المســـتخدمة. ولإجراء عملية التحويل هذه، تســـتخدم أدوات هندســـة البرمجيات خطوات شـــبيهة بخطـــوات التحويـــل التى يتطرق إليها هـــذا الفصل. أما الخطـــوة الثانية من مرحلة التصميـــم المنطقى والمتمثلة فـــى عملية التطبيع فهى محور الجـــزء الأول من الفصل السادس.

٥-١ التحويل من النموذج المفاهيمى كينونة-علاقة إلى النموذج العلاقى:

خـــلال عملية التصميم المنطقى لقاعـــدة البيانات يتم تحويـــل النموذج المفاهيمى كينونـــة - علاقة إلى هياكل قاعدة بيانات علاقية. وتكون المدخلات لهذه العملية هى نموذج بيانات كينونة - علاقة، أما مخرجات هذه العملية فهى هياكل قاعدة البيانات. كما أن هذه العمليه تعد عملية بســـيطة إلى حد ما، ولها قواعدها المعروفة لدرجة أن الكثير من أدوات هندســـة البرمجيات (CASE Tools) تقوم بعملية التحويل هذه بشـــكل آلى، كما أسلفنا أعلاه. إلا أنه من الضرورى التعرف على خطوات هذه العملية لثلاثة أسباب (Hoffer et al, 2002):

١- لا تستطيع معظم أدوات هندسة البرمجيات نمذجة علاقات معقدة مثل العلاقات الثلاثية وعلاقات الأنواع الرئيســـية والأنـــواع الفرعية ومن ثم تحويلها إلى جداول علاقية. فى مثل هذه الحالات قد يتطلب الأمر تحويل مثل هذه العلاقات بشـــكل يدوى.

٢- قد يتوافر عدد من البدائل لتحويل بعض الحالات فى النموذج المفاهيمى التى يمكن اختيار المناسب منها مع الوضع الذى نحاول نمذجته.

٣- قـــد يتطلب الأمر التأكد من جودة مخرجات أدوات هندســـة البرمجيات من حيث إنها قد قامت بتحويل النموذج المفاهيمى إلى جداول جيدة التصميم.

وفيما يلى سنوضح عملية التحويل من النموذج المفاهيمى كينونة - علاقة إلى النموذج العلاقى مستخدمين الأمثلة التى سبق التطرق إليها فى الفصل الثانى والفصل الثالث، مع التركيز على الحالة الدراسية المتعلقة بالجامعة الأهلية.

٥-١-١ قاعدة التحويل الأولى: التعامل مع الكينونات القوية (أو العادية) وخصائصها:

لكل كينونة قوية موجودة ضمن النموذج المفاهيمى كينونة-علاقة يتم إنشاء جدول يحمل مسمى الكينونة، ويحتوى على جميع الخصائص البسيطة المرتبطة بالكينونة باعتبارها حقولاً ضمن الجدول قيد الإنشاء. أما بالنسبة للخصائص المركبة، فيتم إنشاء حقول لمكوناتها البسيطة فقط ضمن الجدول. أما خاصية المعرف فتصبح المفتاح الرئيسى للجدول قيد الإنشاء. وإذا وُجدت خاصية مشتقة ضمن خصائص كينونة معينة، فإن مثل هذه الخاصية لا يتم إنشاء حقل مقابل لها فى الجدول قيد الإنشاء؛ وذلك لأنه بالإمكان حساب (أو استخلاص) قيمة هذه الخاصية من خلال الحقول الأخرى التى تم إنشاؤها فى الجدول، إلا أنه يجب الملاحظة أن مثل هذه الخاصية المشتقة يجب تمثيلها وعدم تجاهلها فى النموذج المفاهيمى للإشارة إلى أن قيمتها ذات أهمية، وأنه سيتم حسابها من قبل التطبيقات التى ستتعامل مع قاعدة البيانات مستقبلاً.

ولأنه يوجد لدينا فى النموذج المفاهيمى للجامعة الأهلية أربعة كينونات قوية وهى: كينونة عضو هيئة التدريس (FACULTY)، وكينونة القسم الدراسى (DEPARTMENT)، وكينونة الطالب (STUDENT)، وكينونة المادة الدراسية (COURSE)، فإنه يتم تحويلها إلى أربعة جداول علاقية حسب قواعد تحويل الكينونات القوية أعلاه. والشكل رقم (٥-١) يوضح عملية تحويل الكينونات الأربع، مع ملاحظة وضع خط متصل تحت مسمى الحقل الذى يمثل المفتاح الرئيسى لكل جدول.



شكل رقم (٥-١): تحويل الكينونات القوية إلى علاقات (أو جداول)

FName, LName, Name, DOB, Salary, Faculty_ID, FACULTY, Phone_No

كينونة أعضاء هيئة التدريس

FACULTY

Faculty_ID	FName	LName	Phone_No	DOB	Salary

علاقة (أو جدول) أعضاء هيئة التدريس

Name, Department_ID, DEPARTMENT

كينونة الأقسام الدراسية

DEPARTMENT

Department_ID	Name

علاقة (أو جدول) الأقسام الدراسية

FName, LName, Name, Street, Street, Address, Street, Student_ID, STUDENT

كينونة الطلبة

STUDENT

Student_ID	FName	LName	Street	City	Zip-Code

علاقة (أو جدول) الطلبة

Title, Units, Course_ID, COURSE

كينونة المواد الدراسية

COURSE

Course_ID	Title	Units

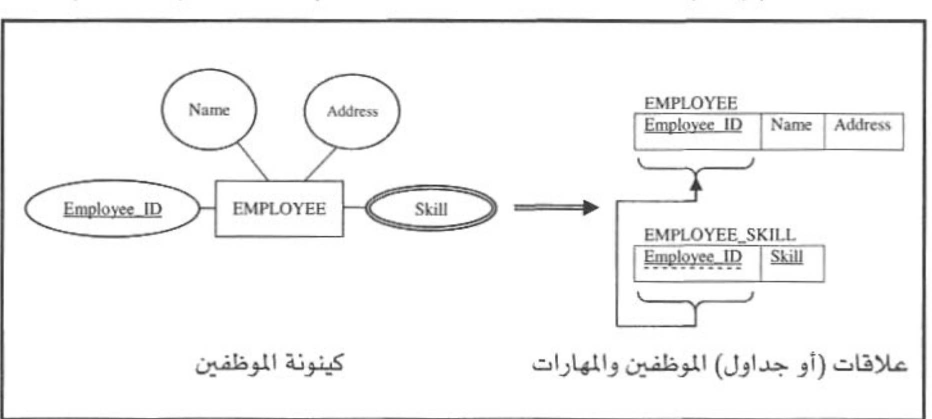
علاقة (أو جدول) المواد الدراسية

٥-١-١-١ التعامل مع الخاصية متعددة القيم:

فــى حال تضمنت إحــدى الكينونات القوية خاصية متعددة القيم فإنه يتم إنشــاء جدولين عوضاً عن جدول واحد بحيث يسمى الجدول الأول باسم الكينونة قيد التحويل، وتكــون حقوله ممثلة لكافة خصائص الكينونة ما عــدا الخاصية المتعددة القيم. أما الجــدول الثانى فيتكون من حقلين مجتمعين يمثلان المفتاح الرئيســى للجدول بحيث يكون أحدهما ممثلاً للمفتاح الرئيســى للجدول الأول، ويكون فى الوقت نفسه مفتاحاً خارجياً يشــير إلى الجدول الأول. أما الحقل الثانى فيمثل الخاصية المتعددة القيم. ويستمد الجدول الثانى مسماه من معنى الخاصية المركبة (أو اسمها الفعلى).

ويوضح الشـــكل رقم (٥-٢) كينونة «الموظف» (EMPLOYEE) التى تتضمن خاصية متعددة القيم وهى خاصية «المهارة» (Skill) للدلالة على أنه قد يكون للموظف الواحد أكثر من مهارة (مثل البرمجة بلغة باسكال، وكوبول، وسى، إلخ). وباتباع قواعد التحويل الســـابقة، يتم إنشـــاء جدولين. الجدول الأول يحمل مسمى «موظف» (EMPLOYEE) ويحتوى على حقول تمثل جميع خصائص كينونة «الموظف» ما عدا الخاصية المتعددة القيم. أما الجدول الثانى واســـمه «مهارة الموظـــف» (EMPLOYEE_SKILL) فيحتوى علـــى حقلين: الأول منهما هو المفتاح الرئيســـى للجـــدول الأول (EMPLOYEE) وهو (Employee_ID)، والثانـــى فيمثل المهارة ويحمل مســـمى «مهـــارة» (Skill). ويمثل كلا الحقلـــين مدمجين أحدهما مع الآخر المفتاح الرئيســـى للجدول؛ إذ يتم توضيح ذلك مـــن خلال وضع خط متصل تحـــت كل منهما. أما الخـــط المتقطع تحت حقل «رقم الموظـــف» (Employee_ID) فى الجـــدول فهو للدلالة على أن هـــذا الحقل يمثل أيضاً مفتاحاً خارجياً يشـــير إلى جـــدول «الموظف»، بالإضافة إلى كونه جـــزءاً من المفتاح الرئيسى للجدول. وقد تم إيضاح عملية الارتباط هذه (بين المفتاح الخارجى والمفتاح الرئيسى)، فى الشكل رقم (٥-٢)، من خلال السهم الواصل بين الحقلين. ويحتوى كل صف فى جدول «مهارة الموظف» على رقم الموظف والمهارة التى تتوافر لديه.

شكل رقم (٥-٢): تحويل الخاصية المتعددة القيم إلى علاقة (أو جدول)



كما يمكن أن يقترح التمثيل السابق على المستفيدين من قاعدة البيانات إضافة المزيد مـــن الحقول إلـــى جدول «مهارة الموظف» مثل تاريخ اكتســـاب المهارة، أو مكان الحصول

عليهـــا،... إلخ. وتجدر الإشـــارة هنا إلى أن الخاصية المتعـــددة القيم قد تكون مركبة أيضاً. فعلى ســـبيل المثال، من الممكن تمثيل خاصيـــة المهارة فى النموذج المفاهيمى من الأساس على أنها تتكون من اسم المهارة، وتاريخ الحصول عليها، ومكان الحصول عليها. فى مثل هذه الحالة، يتم وضع المهارة (Skill) ضمن شـــكل بيضوى مزدوج الخطوط (كما هو أعلاه)، ويتفرع منه بقية الخصائص البسيطة الثلاث موضوعة، كلٌ على حدة، ضمن أشكال بيضوية مفردة الخطوط كما لو كنا نحاول نمذجة خاصية مركبة. وعند ارتباط خاصية مركبة متعددة القيم، يتم تمثيل خصائصها البسيطة فقط ضمن الجدول الثانى، بالإضافة إلى حقل المفتاح الرئيســـى للجدول الأول، كما هو موضح فى الشكل رقم (٥-٣)، على افتراض أن اسم المهارة يعد مميزاً للمهارات التى يتمتع بها الموظفون.

شكل رقم (٥-٣): تحويل الخاصية المركبة المتعددة القيم إلى علاقة (أو جدول)

Name, Address, Employee_ID, EMPLOYEE, Skill, Name, Date, Place

EMPLOYEE

Employee_ID	Name	Address

EMPLOYEE_SKILL

Employee_ID	Name	Date	Place

كينونة الموظفين — علاقات (أو جداول) الموظفين والمهارات

وفى حالة ارتباط أكثر من خاصية واحدة متعددة القيم بكينونة ما، فإنه يتم إنشاء جدول لكل واحدة من الخصائص المتعددة القيم. ويكون المفتاح الرئيسى للجدول الذى يمثل الكينونة الرئيســـية جزءاً من المفاتيح الرئيســـية للجداول التى تمثل الخصائص المتعـــددة القيم. كما يكون جزء المفتاح الرئيســـى للكينونة الرئيســـية مفتاحاً خارجياً فى كل جدول من جداول الخصائص المتعددة القيم يشـــير إلى الجدول الرئيسى. أما الخاصية المتعددة القيم نفسها، فى كل جدول، فتمثل الحقل الثانى فى الجدول، وتكون جزءاً من مفتاحه الرئيسى.

٥-١-٢ قاعدة التحويل الثانية: التعامل مع الكينونات الضعيفة:

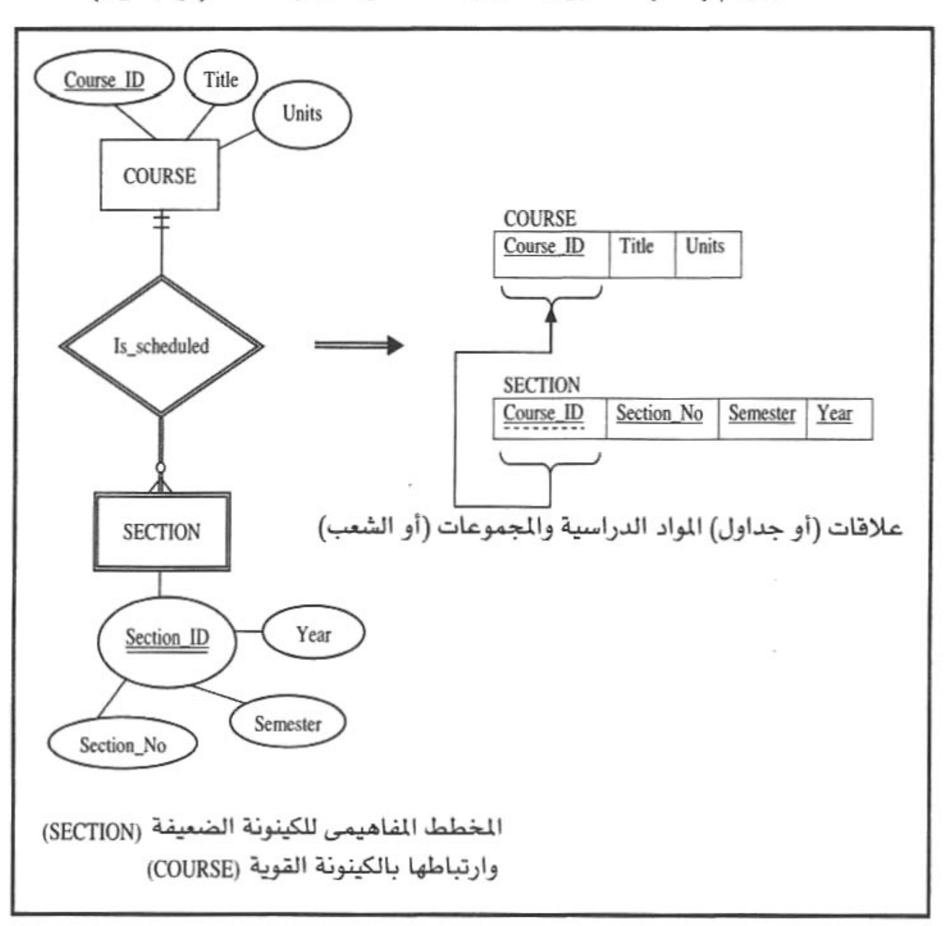
الكينونات الضعيفة هى تلك الكينونات التى لا يمكن أن توجد فى النموذج المفاهيمى

باستقلالية؛ لكونها تعتمد على علاقات معرفة تربطها بكينونات قوية. كما أن أى كينونة ضعيفة لا يتوافر لديها معرف كامل يميز بين حالاتها المختلفة، إلا أنه لا بد أن يتوافر لديها معرف جزئى يتكون من خاصية (أو أكثر) تستطيع أن تميز بين مجموعة الحالات التى ترتبط بكل حالة من حالات الكينونة القوية. والشكل رقم (٥-٤) يوضح ارتباط الكينونة الضعيفة وهى كينونة «المجموعة الدراسية» (SECTION) من خلال العلاقة المعرفة وهى «جدولة المجموعة» (Is_scheduled) بالكينونة القوية، وهى «المادة الدراسية» (COURSE). ومعنى هذا أنه لا يمكن أن توجد مجموعة دراسية دون أن ترتبط بمادة دراسية معينة ضمن كينونة «المادة الدراسية». كما أن الخصائص البسيطة المكونة للخاصية المركبة «رمز المجموعة» (Section_ID) يمكننا من التمييز بين المجموعات الدراسية التابعة لأى مادة دراسية، ولكنها لا تستطيع أن تميز بين المجموعات التابعة لمواد دراسية مختلفة لكونها قد تأخذ القيم نفسها. لذلك فإن الخاصية المركبة تعد مميزاً جزئياً للكينونة الضعيفة. لذلك فإننا نستخدم، عند تحويل الكينونة الضعيفة لجدول علاقى، الخاصية المعرفة للكينونة القوية بالإضافة إلى المعرف الجزئى للكينونة الضعيفة فى تعريف المفتاح الرئيسى للجدول.

وتتم عملية تحويل أى كينونة ضعيفة إلى النموذج العلاقى من خلال إنشاء جدول يحتوى على حقول لكافة الخصائص البسيطة المرتبطة بالكينونة، كما لو أننا نقوم بتحويل كينونة قوية إلى جدول علاقى. بالإضافة إلى ذلك يتم إدراج حقل فى الجدول لخاصية معرف الكينونة القوية. ويكون المفتاح الرئيسى لجدول الكينونة الضعيفة عبارة عن حقل (أو حقول) المعرف الجزئى للكينونة الضعيفة، بالإضافة إلى حقل (أو حقول) المفتاح الرئيسى للكينونة القوية. فعلى سبيل المثال، عند تحويل كينونة «المجموعة الدراسية» إلى النموذج العلاقى، يتم إنشاء جدول بمسمى الكينونة نفسه ويحتوى على حقول لتمثيل جميع الخصائص البسيطة المرتبطة بالكينونة وهى: «رقم المجموعة»، و«الفصل» الدراسى المنفذة فيه، و«السنة» الدراسية المنفذة فيها. ولكون هذه الحقول الثلاثة مجتمعة تعد معرفاً جزئياً لا يمكننا من التمييز بين المجموعات التابعة لمواد دراسية مختلفة، فقد تم إدراج حقل لتمثيل معرف الكينونة القوية وهو «رمز المادة الدراسية»، وفى ذات الوقت تم تعريف هذا الحقل على أساس أنه جزء من المفتاح الرئيسى للجدول. وبهذه الطريقة يمكننا الآن التمييز بين جميع المجموعات الدراسية، بشكل منفرد، بغض النظر عن المادة الدراسية التى تتبع لها. ولأن وجود كينونة ضعيفة يعنى دائماً وجود علاقة بينها وبين الكينونة القوية التى ترتكز عليها وأن

هذه العلاقة لا يمكن أن تكون متعدد - متعدد (لأن كل حالة من حالات الكينونة الضعيفة لا يمكــن أن ترتبــط بأكثر من حالة من حالات الكينونة القويــة)، فإنه يتم تمثيل هذه العلاقة من خلال تعريف حقل «رمز المادة الدراســية» فى جدول المجموعات الدراسية على أنه مفتاح خارجى يشير إلى المفتاح الرئيسى فى جدول المواد الدراسية. وقد تم إيضاح ذلك فى الشكل رقم (٥-٤) من خلال وضع خط متقطع تحت حقل «رمز المادة الدراسية»، بالإضافة إلى استخدام ســهم يوضح عملية الارتباط هذه بين الجدولين. وسيتم شرح طرق تحويل العلاقات بشكل أكثر تفصيلاً فى الأجزاء التالية.

شكل رقم (٥-٤): تحويل الكينونة الضعيفة إلى علاقة (أو جدول)



علاقات (أو جداول) المواد الدراسية والمجموعات (أو الشعب)

المخطط المفاهيمى للكينونة الضعيفة (SECTION) وارتباطها بالكينونة القوية (COURSE)

٥-١-٣ قاعدة التحويل الثالثة: التعامل مع العلاقات الثنائية:

تعتمد عملية تحويل العلاقات على درجاتها من حيث كونها أحادية، أو ثنائية، أو ثلاثية (فأكثر). كما أن عملية التحويل تعتمد أيضاً على تعددية العلاقة من حيث كونها واحد - واحد، أو واحد - متعدد، أو متعدد - متعدد. وبخلاف تعددية العلاقة فى النموذج المفاهيمى كينونة - علاقة، فإننا ننظر فقط إلى التعددية العليا للعلاقة فى مرحلة التصميم المنطقى، دون النظر إلى التعددية الدنيا لها فى هذه المرحلة. إلا أنه من الضرورى الإشارة إلى أن التعددية الدنيا ذات أهمية كبيرة فى أثناء عملية بناء هياكل قاعدة البيانات كما سنوضح عند شرحنا للغة الاستفسار البنائية فى الفصل السابع والفصل الثامن. وتوضح الأجزاء التالية الطرق المتبعة لتحويل العلاقات حسب درجاتها وتعددياتها.

٥-١-٣-١ التعامل مع العلاقات الثنائية ذات التعددية واحد - متعدد:

عند وجود علاقة ذات درجة ثنائية (تربط بين كينونتين) وتعددية واحد - متعدد، يتم أولاً إنشاء جدول لكل كينونة من الكينونات المرتبطة بالعلاقة الثنائية، وفقاً لقواعد الخطوة الأولى أعلاه. بعد ذلك يتم إدراج المفتاح الرئيسى (سواء كان حقلاً واحداً أو أكثر) للجدول الذى يمثل الكينونة فى الجانب ذى التعددية «واحد» ضمن حقول الجدول الذى يمثل الكينونة فى الجانب ذى التعددية «متعدد». ويتم تعريف هذا الحقل (أو الحقول) باعتباره مفتاحاً خارجياً يشير إلى الجدول الذى يمثل الكينونة فى الجانب ذى التعددية «واحد».

ولإيضاح عملية التحويل هذه، لنأخذ على سبيل المثال العلاقة الثنائية «يعمل فى» (Works_for) التى تربط بين كينونة «الأقسام الدراسية» (DEPARTMENT) وكينونة «أعضاء هيئة التدريس» (FACULTY) فى نموذج كينونة - علاقة للجامعة الأهلية الموضحة فى الشكل رقم (٥-٥). فهذه العلاقة، بالإضافة إلى كونها علاقة ثنائية، فهى علاقة ذات تعددية واحد - متعدد، وذلك لكون كل عضو هيئة تدريس يعمل فى قسم دراسى واحد (على الأكثر) وأن كل قسم دراسى يعمل فيه أكثر من عضو هيئة تدريس. ولأن التعددية «واحد» فى هذا النموذج تأتى فى جانب كينونة الأقسام الدراسية، فإنه يتم إدراج حقل جديد فى جدول أعضاء هيئة التدريس، ذى الجانب المتعدد، لتمثيل المفتاح الرئيسى لجدول الأقسام العلمية. ويتم تعريف هذا الحقل على

أنه مفتاح خارجى. وقد تم إيضاح ذلك فى عملية التحويل من خلال وضع خط متقطع تحت مســمى هذا الحقل، كما تم وضع سهم يشير إلى هذا الارتباط. وبهذه الطريقة يمكننا دائماً معرفة القسم الذى يعمل فيه كل عضو من أعضاء هيئة التدريس. ونظراً لكون التعددية إجبارية فإن حقل رمز القسم الدراسى فى جدول أعضاء هيئة التدريس لا يمكن أن يكون غير معرفٍ (NULL). وعلى الرغم من عدم إمكانية فرض هذا القيد على حقل المفتاح الخارجى فى هذه المرحلة من التصميم، إلا أنه يمكن فرضه فى أثناء مرحلة بناء قاعدة البيانات باســتخدام قيد الحقول (NOT NULL) كما ســنوضح فى الفصل السابع.

شكل رقم (٥-٥): تحويل العلاقة الثنائية ذات التعددية واحد - متعدد إلى النموذج العلاقى

Department_ID
Name
DEPARTMENT
Works_for
Salary
DOB
FACULTY
Phone_No
Faculty_ID
Name
FName
LName

DEPARTMENT

Department_ID	Name

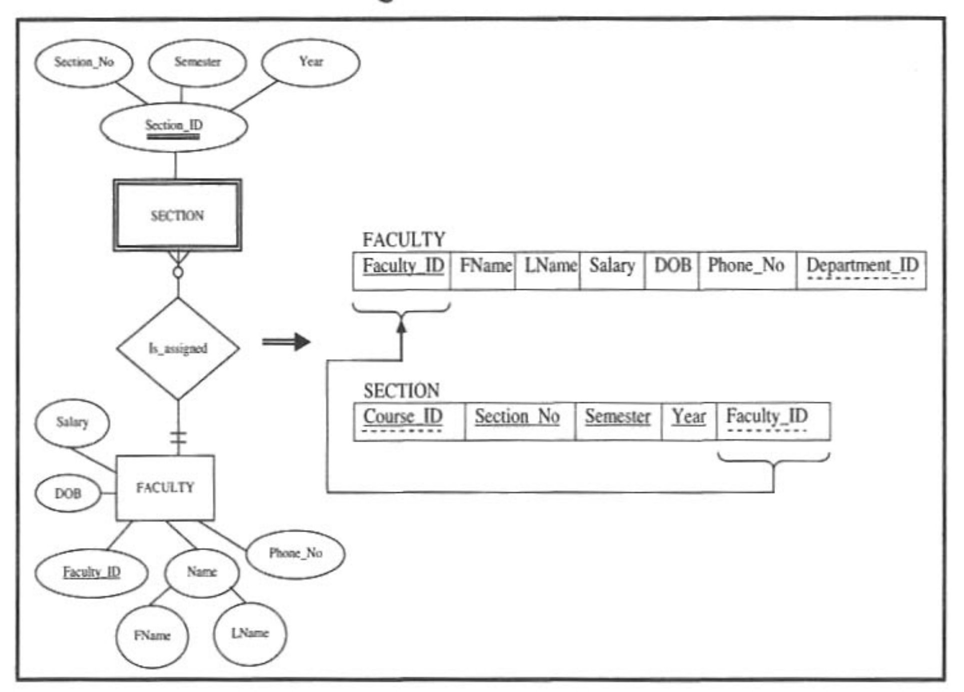
FACULTY

Faculty_ID	FName	LName	Salary	DOB	Phone_No	Department_ID

وقد يُطرح الســؤال التالى: هل من الممكن وضع المفتاح الرئيســى لجدول أعضاء هيئــة التدريس باعتبــاره مفتاحاً خارجياً فى جدول الأقســام العلمية لتمثيل العلاقة الثنائية السابقة فى النموذج العلاقى عضواً عن تمثيلها بالطريقة السابقة؟ إن الإجابة عن التســاؤل هى عدم إمكانية ذلك، والسبب يرجع إلى أنه لو فعلنا ذلك فإننا سنقوم بتكرار قيمة المفتاح الرئيسى واسم القسم لجميع أعضاء هيئة التدريس الذين يعملون فى القســم الدراسى نفســه، وبذلك لن يصبح حقل رمز القســم، فى جدول الأقسام

العلميـــة مفتاحاً رئيســياً؛ لكونه يتكرر فى ســجلات الجدول. كمــا أن عملية تكرار البيانات هذه قد تؤدى إلى إشــكالات (Anomalies) عند التعديل على البيانات يصعب فى ظل وجودها التحكم فى تناسق البيانات (Data Consistency)، كما سيتضح عن شرح الجداول جيدة البنــاء وعمليات تطبيع الجداول (Normalization) (فى الجزء الأول من الفصل السادس).

شكل رقم (٥-٦): تحويل العلاقة الثنائية ذات التعددية واحد - متعدد ترتبط بها كينونة ضعيفة إلى النموذج العلاقى



أما الشكل رقم (٥-٦) فيمثل علاقة ثنائية واحد - متعدد تربط بين كينونة قوية، وهى كينونة «أعضاء هيئــة التدريس» (FACULTY) وكينونة ضعيفة هى كينونة «المجموعات الدراســية» (SECTION). وتعنى هذه العلاقــة، الممثلة فى النموذج، أن كل عضو هيئة تدريس قد يدرس مجموعة دراسية أو أكثر فى حين تدرس كل مجموعة دراسية من قبل عضو هيئة تدريس واحد فقط. ونظراً لأن العلاقة بين الكينونتين ليست علاقة معرفة (Identifying Relationship)، بمعنــى أن كينونة «أعضاء هيئة التدريس» ليســت الكينونة

التى تميز بين المجموعات الدراسية، وإنما كينونة «المواد الدراسية» هى الكينونة القوية التى تميز بين المجموعات المختلفة، فإنه يتم التعامل مع الكينونة الضعيفة كأنها كينونة قوية عند عملية التحويل للنموذج العلاقى، وذلك حسب القاعدة السابقة. ففى هذه الحالة يتم إدراج حقل جديد ضمن جدول «المجموعات الدراسية» (وهو الجانب المتعدد) لتمثيل المفتاح الرئيسى لجدول «أعضاء هيئة التدريس»، كما يتم تعريف هذا الحقل على أنه مفتاح خارجى من خلال وضع خط متقطع تحت مسماه. كما يلاحظ وجود حقول أخرى فى الشكل قد تم وضع خطوط متقطعة تحت مسمياتها، سواء فى جدول «أعضاء هيئة التدريس» أم فى جدول «المجموعات الدراسية». وهذه الحقول تظهر بهذا الشكل للدلالة على أنها مفاتيح خارجية تمثل علاقات أخرى مع هذين الجدولين تم تحويلها فى المراحل السابقة.

٥-١-٣-٢ التعامل مع العلاقات الثنائية ذات التعددية متعدد - متعدد:

عند وجود علاقة ثنائية ذات تعددية متعدد - متعدد تربط بين كينونتين فإننا نقوم بإنشاء ثلاثة جداول، اثنان منها لتمثيل كل كينونة على حدة (حسب القاعدة (١) أعلاه)، وجدول ثالث لتمثيل العلاقة نفسها بحيث يتضمن جدول العلاقة حقلين يمثلان المفاتيح الرئيسية لجدولى الكينونتين، بالإضافة إلى الحقول اللازمة التى تمثل الخصائص المرتبطة بالعلاقة نفسها (عند وجود مثل هذه الخصائص). ويصبح حقلا المفاتيح الرئيسية لجدولى الكينونتين التى تربط بينهما العلاقة مجتمعين المفتاح الرئيسى للجدول الذى يمثل العلاقة، كما يتم تعريف كل منهما باعتباره مفتاحاً خارجياً يشير إلى الكينونة التى تم جلب الحقل منها.

ويمثل الشكل رقم (٥-٧) علاقة ثنائية ذات تعددية متعدد - متعدد وهى علاقة «يعمل على» (Works_on) تربط بين كينونتين هما كينونة «الموظف» (EMPLOYEE) وكينونة «المشروع» (PROJECT). ويمكن أن تقرأ هذه العلاقة كما يلى «يعمل كل موظف على صفر أو أكثر من المشاريع، وكل مشروع يعمل عليه موظف واحد أو أكثر. وعندما يعمل الموظف على مشروع فإن هناك تاريخاً يمثل بداية عمله على المشروع». ونظراً لأن تاريخ العمل على المشروع ليس من خصائص أى من كينونة «الموظف» أو كينونة «المشروع»، وإنما هى من خصائص العلاقة التى تربط بينهما، فقد تم ربطها بالعلاقة وليس بأى من الكينونتين. وحسب قاعدة التحويل أعلاه، يتم إنشاء ثلاثة جداول: جدولان يمثلان الكينونتين التى تربط بينهما العلاقة وهما جدول «الموظف» وجدول

«المشـــروع»، وجدول ثالث يمثل العلاقة نفسها، وقد سمى جدول «العمل» (JOB). وقد تم تغيير مســـمى الجدول هنا عن مســـمى العلاقة حتي يتوافق مسمى جدول العلاقة (أو جدول الربط) مع قواعد تســـمية الكينونات والجداول التى يفضل أن تكون أسماءً عوضاً عن تسميتها بأفعال.

شكل رقم (٥-٧): تحويل العلاقة الثنائية ذات التعددية متعدد - متعدد إلى النموذج العلاقى

ويوضح الشكل رقم (٥-٧) أيضاً الحقول المكونة للجدولين الذين يمثلان الكينونتين اللتين تربط بينهما العلاقة حيث تم تعريفهما حســـب قاعدة التحويل رقم (١) أعلاه. أما فيما يتعلق بجدول العلاقة نفسها فقد تم تعريف حقلين فيه أحدهما لتمثيل المفتاح الرئيسى لجدول «الموظف» والثانى لتمثيل المفتاح الرئيسى لجدول «المشروع». كما تم تعريف حقل ثالـــث فى جدول العلاقة لتمثيل خاصية «بدايـــة العمل» (Starting_Date) المرتبطـــة بالعلاقة نفســـها. كما يلاحظ فى جدول العلاقـــة وضع خط متصل تحت

الحقلين الذين يمثلان المفاتيح الرئيسية لجدولى «الموظف» وجدول «المشروع» للدلالة على أن كليهما مجتمعين يمثلان المفتاح الرئيسى لجدول العلاقة. بالإضافة إلى ذلك فقد تم وضع خط متقطع تحت كل من الحقلين الذين يمثلان المفاتيح الرئيسية لجدول «الموظف» وجدول «المشروع» فى جدول العلاقة، وذلك للدلالة على أن كلاً منهما يمثل مفتاحاً خارجياً يشير إلى الجدول الذى جلب منه. ولإيضاح ذلك فقد تم استخدام أسهم تصل بين كل مفتاح خارجى بحقل الجدول الذى يشير إليه المفتاح.

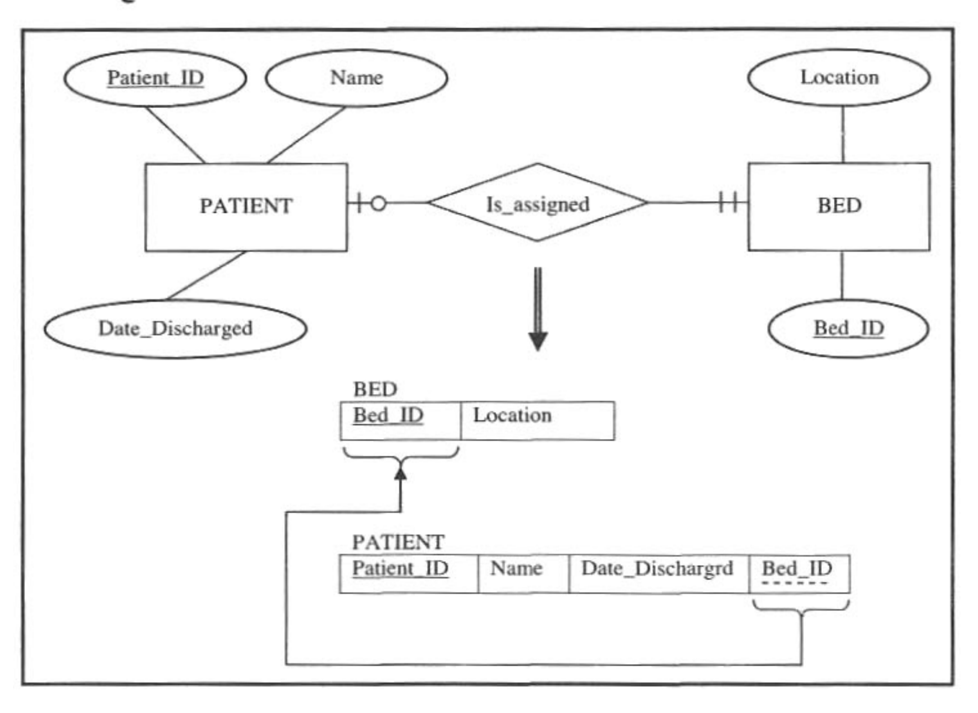
إلا أنه قد يُطرح السؤال التالى: لماذا لا يتم تمثيل العلاقة الثنائية متعدد - متعدد من خلال إضافة حقل لأحد جدولى الكينونتين، ويصبح هذا الحقل مفتاحاً خارجياً فى الجدول الذى تمت إضافته إليه (لتمثيل العلاقة) كما هو الحال فى العلاقات الثنائية ذات التعددية واحد - متعدد عوضاً عن تعريف جدول ثالث خاص بالعلاقة نفسها؟ والإجابة عن ذلك أن هذه الطريقة لا تمكن من تمثيل العلاقات الثنائية ذات التعددية متعدد - متعدد. والسبب وراء ذلك هو أننا لا نعرف الحد الأعلى من عدد المشاريع التى من الممكن أن يعمل عليها الموظف الواحد أو الحد الأعلى من عدد الموظفين الذين من الممكن أن يعملوا على المشروع الواحد. ونتيجة لذلك لا نستطيع تعريف عدد محدد من الحقول، سواء فى جدول الموظفين أم فى جدول المشاريع لتمثيل العلاقة. وحتى لو عرف الحد الأعلى من عدد المشاريع أو الحد الأعلى من عدد الموظفين، فإن مثل هذا التمثيل سيضيع الكثير من مساحة التخزين؛ لأن عدد الموظفين الذين يعملون على كل مشروع (أو عدد المشاريع التى يعمل عليها كل موظف) قد تتفاوت بشكل كبير. بالإضافة إلى ذلك فإنه يجب تمثيل تاريخ البدء فى العمل على كل مشروع من قبل كل موظف، وفق الحد الأعلى المستخدم، مما سيزيد من حجم المساحة التخزينية المهدرة.

٥-١-٣-٣ التعامل مع العلاقات الثنائية ذات التعددية واحد - واحد:

يمكن النظر إلى العلاقات الثنائية ذات التعددية واحد- واحد على أنها حالة خاصة من العلاقات الثنائية واحد - متعدد إذ يتم تحويلها بخطوتين: فى الخطوة الأولى يتم إنشاء جدول لكل من الكينونتين اللتين تربط بينهما العلاقة الثنائية. أما فى الخطوة الثانية فيتم تمثيل المفتاح الرئيسى لجدول إحدى الكينونتين باعتباره حقلاً فى الجدول الذى يمثل الكينونة الأخرى، ويتم تعريف الحقل المضاف على أنه مفتاح خارجى للجدول الذى يمثل الكينونة الأولى. ويلاحظ مدى تشابه هاتين الخطوتين

مــع قاعدة تحويل العلاقات الثنائية ذات التعدديــة واحد - متعدد. إلا أن الاختلاف هنــا يظهر عند تحديد الجدول الذى ســيضاف إليه المفتاح الخارجى والذى ســيمثل العلاقــة. ففى حالة التعددية واحد - متعدد، يتــم إضافة حقل المفتاح الخارجى فى جدول الجانب المتعدد من العلاقة. أما هنا فيضاف المفتاح الرئيســى لجدول الجانب «الإجبارى» من العلاقة باعتباره مفتاحاً خارجياً فى جدول الجانب «الاختيارى». وكما أســلفنا ســابقاً فإن ارتباط أية كينونة بعلاقة إما أن يكون إجبارياً أو اختيارياً. ويمثل هذا ضمن النموذج المفاهيمى كينونة - علاقة بالقيمة الصغرى (أو الدنيا). فإذا كانت القيمة الصغرى صفراً فإن العلاقة تعد اختيارية. أما إذا كانت القيمة الصغرى واحداً فإنها تعد إجبارية.

شكل رقم (٥-٨): تحويل العلاقة الثنائية ذات التعددية واحد - واحد إلى النموذج العلاقى



وفــى العلاقات الثنائيــة ذات التعددية واحد - واحد، تكــون التعددية فى غالبية الأحيــان إجبارية من جانب واختيارية من الجانب الآخر. فعلى ســبيل المثال، يوضح الشــكل رقم (٥-٨) كينونة «مريض» (PATIENT) وكينونة «سرير» (BED) اللتين ترتبط

إحداهمــا بالأخرى من خلال العلاقة الثنائية واحد- واحد وهى علاقة «يســند إلى» (Is_assigned). ويعنى التمثيل الموضح فى الشــكل أن كل مريض فى المستشــفى يجب أن يســند إلى ســرير واحد فقط، فى حين قد يسند السرير لمريض ما أو قد لا يسند لأى مريــض. وبناءً على هذا التمثيل فإن الجانب الإجبارى هو من جهة الســرير؛ إذ إن كل مريض لا بد أن يســند إلى ســرير. وباتباع القاعدة أعلاه، يتم إنشاء جدولين أحدهما لتمثيل كينونة «المريض» والآخر لتمثيل كينونة «السرير». كما تتم إضافة حقل جديــد فى جدول الجانب الاختيارى من العلاقة، وهو جدول «المريض»، لتمثيل المفتاح الرئيســى لجدول الجانب الإجبارى، وهو «السرير». كما يتم تعريف الحقل الذى تمت إضافته على أنه مفتاح خارجى يشــير إلى جدول الجانب الإجبارى، وهو «الســرير». وفى حال ارتبطت العلاقة نفسها بخصائص فإنها تضاف أيضاً باعتبارها حقولاً ضمن الجانب الاختيارى من العلاقة. فلو ارتبطت علاقة إســناد السرير فى مثالنا بخاصية مثل «تاريخ بداية الإســناد» (Starting_Date)، فإنــه يتم تمثيل هذه الخاصية باعتبارها حقلاً مصاحباً للمفتاح الخارجى ضمن جدول «المريض».

والســؤال الذى قد يُطرح هو: هل بالإمكان تمثيل العلاقة الثنائية ذات التعديدية واحــد - واحد بالشــكل المعاكــس؟ بمعنى هل من الممكن أن يعرف المفتاح الرئيســى لجــدول الجانب «الاختياري» (وهو المريض فى مثالنــا) على أنه مفتاح خارجى ضمن جدول الجانب الإجبارى (وهو «الســرير» فى مثالنا، بالإضافة لأية خصائص قد تكون مرتبطة بالعلاقة نفسها)؟ إن الإجابة عن هذا التساؤل هى: نعم يمكننا ذلك، ولكنها ليســت الطريقة المثلى. والسبب وراء ذلك أن «الســرير» ليس من الضرورى أن يكون مســنداً إلى أى مريض. وفى هذه الحالة ســيتم وضع قيمة غير معرفة (NULL) فى حقــل المفتاح الخارجى (وبقية الحقول المرتبطة بالعلاقــة إن وجدت) للدلالة على أن الســرير غير مســند إلى أى مريض. ويعنى هذا إهدار المساحة التخزينية فى حالات عدم الإسناد هذه.

علــى الرغم من أن الحالة العامة للعلاقات الثنائية ذات التعددية واحد- واحد هى وجود جانب إجبارى وجانب اختيارى، إلا أنه من الممكن أن يكون كلا الجانبين إجباريين أو كلاهمــا اختياريين. ففى الحالة الأولى (وهى كــون العلاقة إجبارية من الجهتين) تتم عملية تحويل العلاقة حسب الخطوتين السابقتين، وبحيث يتم تعريف حقل المفتاح الرئيســى لأى من الجدولين على أنه مفتاح خارجــى ضمن الجدول الآخر (بالإضافة إلى الحقول المرتبطة بالعلاقة) دون أية أفضلية بين الجدولين. أما فى الحالة الثانية (وهى كون العلاقة اختيارية من الجهتين)، وعلى الرغم من إمكانية اســتخدام طريقة

التحويل السابقة نفسها، إلا أن عملية التحويل يمكن أن تتم كما لو أننا نقوم بتحويل علاقة ثنائية ذات تعددية متعدد - متعدد، بمعنى إنشاء جدول ثالث لتمثيل العلاقة عوضاً عن تضمينها ضمن أحد الجدولين. وتأتى أهمية التمثيل الأخير للعلاقات الثنائية ذات التعددية واحد - واحد عند كون حالات عدم ارتباط الكينونتين التى تربط بينهما العلاقة هى الحالة العامة، إذ إنه سيتم تجنب الكثير من وجود القيم غير المعرفة مقارنة بالطريقة السابقة.

٥-١-٤ قاعدة التحويل الرابعة: التعامل مع الكينونات المشاركة:

إن الكينونة المشاركة هى فى أصلها علاقة ذات تعددية متعدد - متعدد، ولكن الشخص الذى يقوم بتصميم نموذج كينونة - علاقة قد يرى أنه من الأنسب تمثيل هذه العلاقة باعتبارها كينونة مشاركة، وذلك عندما يكون تمثيلها بهذه الطريقة أقرب إلى فهم المستفيدين من قاعدة البيانات عوضاً عن تمثيلها كعلاقة ذات تعددية متعدد - متعدد. أما عملية تحويل الكينونة المشاركة من النموذج المفاهيمى إلى النموذج العلاقى فهى مماثلة لعملية تحويل العلاقات ذات التعددية متعدد - متعدد. وتتكون عملية التحويل من خطوتين: فى الخطوة الأولى يتم تعريف ثلاثة جداول: اثنان منها لتعريف الكينونتين اللتين تربط بينهما الكينونة المشاركة، والثالث لتعريف الكينونة المشاركة نفسها. أما الخطوة الثانية فتعتمد على وجود معرف للكينونة المشاركة من عدم وجود معرف لها، كما يلى:

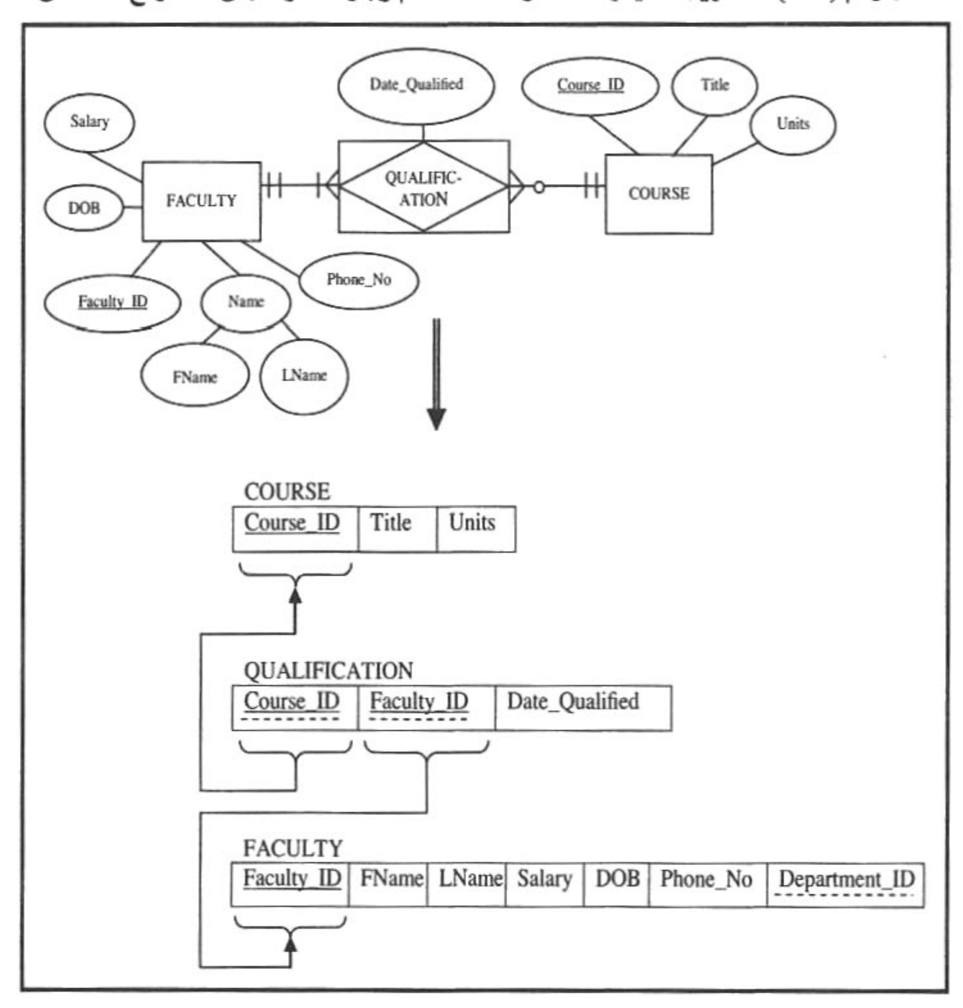
٥-١-٤-١ التعامل مع الكينونات المشاركة عند عدم وجود معرف:

عند عدم ارتباط الكينونة المشاركة بخاصية معرفة تميز بين حالات الكينونة المشاركة، يتم استخدام المفاتيح الرئيسية للجدولين اللذين يمثلان الكينونتين اللتين تربط بينهما الكينونة المشاركة مجتمعين باعتبارها مفتاحاً رئيسياً لجدول الكينونة المشاركة. وفى الوقت نفسه يتم تعريف كل واحد من هذين المفتاحين الرئيسيين على أنه مفتاح خارجى يشير إلى جدول الكينونة التى جلب منها. وبهذه الطريقة تأتى عملية تحويل العلاقة المشاركة متطابقة مع عملية تحويل العلاقات ذات التعددية متعدد - متعدد.

ويوضح الشكل رقم (٥-٩) عملية تحويل إحدى الكينونات المشاركة الموجودة فى نموذج كينونة - علاقة للجامعة الأهلية. ويحتوى الشكل على كينونة مشاركة هى علاقة «تأهيل» (QUALIFICATION) تربط بين كينونة «عضو هيئة التدريس» (FACULTY) وكينونة «المادة الدراسية» (COURSE). كما يرتبط بالعلاقة المشاركة خاصية واحدة

هى خاصية «تاريـــخ التأهيل» (Date_Qualified) تحدد التاريخ الذى تم فيه تأهل عضو هيئة التدريس لتدريس مادة ما.

شكل رقم (٥-٩): تحويل الكينونة المشاركة عند عدم وجود معرف إلى النموذج العلاقى



وباتباع خطوتى التحويل أعلاه، يتم إنشـــاء ثلاثة جـــداول هي: جدول «أعضاء هيئة التدريس» (FACULTY) لتمثيل كينونة «أعضاء هيئة التدريس»، وجدول «المواد الدراسية» (COURSE) لتمثيل كينونة «المواد الدراسية»، وجدول «تأهيل» (QUALIFICATION) لتمثيل الكينونة المشاركة «تأهيل». ونظراً لعدم ارتباط الكينونة المشاركة بخاصية معرفة، فإنه

يتـم تعريف المفتاح الرئيسـى لجدول «أعضـاء هيئة التدريس» وهـو «رقم عضو هيئة التدريـس» (Faculty_ID)، والمفتاح الرئيسـى لجدول «المواد الدراسـية» وهو «رقم المادة الدراسية» (Course_ID)، مجتمعين، كمفتاح رئيسى للكينونة المشاركة. وفى الوقت نفسه، يتـم تعريف كل جزء من المفتاح الرئيسـى فى جدول الكينونة المشـاركة على أنه مفتاح خارجى يشير إلى أحد جدولى الكينونتين اللتين تربط بينهما الكينونة المشاركة. فالحقل «رقم المادة الدراسـية» (Course_ID) فى جدول «تأهيل» (QUALIFICATION) يمثل جزءًا من المفتاح الرئيسـى للجدول، وفى الوقت نفسـه يمثل مفتاحـاً خارجياً لجدول «المواد الدراسـية» (COURSE). كذلك هو الحال بالنسـبة لحقل «رقـم عضو هيئة التدريس» (Faculty_ID) الذى يمثل جزءًا من المفتاح الرئيسـى لجدول «التأهيل»، وفى الوقت نفسه يمثل مفتاحاً خارجياً يشـير إلى جدول «أعضاء هيئة التدريس». وقد تم إيضاح المفتاح الرئيسـى لجدول «التأهيل» من خـلال وضع خط متصل تحت الحقلـين الذين يتكون منهمـا، وإيضاح المفاتيح الخارجية من خلال وضع خط متقطع تحت كل منهما. كما تم إيضاح الجدول الذى يشير إليه كل مفتاح خارجى من خلال السهم الذى يصل بين المفتاح والجدول الذى يشير إليه المفتاح.

٥-١-٤-٢ التعامل مع الكينونات المشاركة عند وجدود معرف:

فى بعض الأحيان تكون الكينونة المشاركة مرتبطة بخاصية معرفة تميز بين حالات العلاقة المشاركة. وهناك سببان يحفزان ربط الكينونة المشاركة بمعرف وهما (Hoffer et al, 2002):

١- قد يكون من الطبيعى وجود معرف للكينونة المشاركة معروف من قبل المستخدمين لقاعدة البيانات.

٢- اسـتخدام المفاتيح الرئيسية لجداول الكينونات التى تربط بينها الكينونة المشاركة على أنها مفتاح رئيسى فى جدول الكينونة المشاركة قد لا يمكّن من تمييز الحالات فى جدول الكينونة المشاركة بشكل منفرد.

ولتحويل الكينونة المشـاركة التى ترتبط بمعـرف إلى النموذج العلاقى، يتم تعريف جـدول خاص بالكينونة المشـاركة، كما سـبق أعـلاه، بالإضافة إلـى تعريف جداول الكينونات التى تربط بينها الكينونة المشاركة. إلا أن وجه الاختلاف هنا يكمن فى أن المفتاح الرئيسـى لجدول الكينونة المشاركة سيكون معرف الكينونة عوضاً عن المفاتيح الرئيسية لجداول الكينونات التى تربط بينها الكينونة المشاركة. أما المفاتيح الرئيسية

لجـــداول الكينونـــات التى تربط بينها الكينونة المشـــاركة فيتـــم تعريفها ضمن جدول الكينونة المشاركة باعتبارها مفاتيح خارجية.

شكل رقم (٥-١٠): تحويل الكينونة المشاركة عند وجود معرف إلى النموذج العلاقى

ويوضـــح الشـــكل رقـــم (٥-١٠) عمليـــة تحويل إحـــدى الكينونات المشـــاركة التى ترتبط بخاصية معرفة. ويحتوى الشـــكل على كينونة مشـــاركة هى علاقة «إرسالية»

(SHIPMENT) تربط بين كينونة «العميل» (CUSTOMER) وكينونة «مورد» (VENDOR). كما يرتبط بالكينونة المشاركة ثلاث خصائص من ضمنها الخاصية المعرفة «رقم الإرسالية» (Shipment_NO). وقد تم تحديد هذه الخاصية معرفاً للكينونة المشاركة لسببين:

١- يعد «رقم الطلبية» معرفاً طبيعياً للكينونة المشاركة متعارفاً عليه فى بيئة المستفيدين.

٢- لا يمكن أن تعرف خاصية «رقم العميل» المرتبطة بكينونة «العميل» وخاصية «رقم المورد» المرتبطة بكينونة «المورد» حالات الكينونة المشاركة بشكل منفرد؛ وذلك لأن المورد الواحد قد يرسل أكثر من إرسالية للعميل نفسه. وحتى لو تم استخدام بقية خصائص الكينونة المشاركة (وهى التاريخ والكمية) بالإضافة للخاصيتين السابقتين كمعرف للكينونة المشاركة، فإنه لا يمكن التيقن من أن هذه الخصائص مجتمعة ستتمكن من تمييز حالات الكينونة المشاركة بشكل منفرد. والسبب وراء ذلك هو أن المورد الواحد قد يرسل لنفس العميل أكثر من إرسالية واحدة بنفس التاريخ وبنفس الكمية.

وباتباع خطوتى التحويل أعلاه، يتم إنشاء ثلاثة جداول وهى: جدول «العميل» (CUSTOMER) لتمثيل كينونة «العميل»، وجدول «المورد» (VENDOR) لتمثيل كينونة «المورد»، وجدول «إرسالية» (SHIPMENT) لتمثيل العلاقة المشاركة «إرسالية». ونظراً لارتباط الكينونة المشاركة بخاصية معرفة وهى «رقم الإرسالية» (Shipment_NO)، فإنه يتم تعريف خاصية المعرف على أنها المفتاح الرئيسى لجدول الكينونة المشاركة. كما يتم تعريف المفتاح الرئيسى لجدول كينونة «العميل» وهو «رقم العميل» (Customer_ID) والمفتاح الرئيسى لجدول كينونة «المورد»، وهو «رقم المورد» (Vendor_ID) كحقول ضمن جدول «إرسالية». ويتم تعريف كل منهما على أنه مفتاح خارجى. كما يتم تعريف بقية خصائص العلاقة المشاركة، وهى «التاريخ» (Date) و«الكمية» (Amount) على أنها حقول ضمن جدول الكينونة المشاركة. وقد تم إيضاح المفتاح الرئيسى لجدول «إرسالية» من خلال وضع خط متصل تحته، وإيضاح المفاتيح الخارجية من خلال وضع خط متقطع تحت كل منهما. كما تم إيضاح الجدول الذى يشير إليه كل مفتاح خارجى من خلال السهم الذى يصل بين المفتاح والجدول الذى يشير إليه المفتاح.

٥-١-٥ قاعدة التحويل الخامسة: التعامل مع العلاقات الأحادية:

العلاقـة الأحاديـة هى علاقـة تربط بين حالات الكينونة نفسـها. وتسـمى هذه العلاقـات فى بعض الأحيان بالعلاقات المتواتـرة (Recursive Relationships). ومن أهم الحـالات التى تظهر فيها العلاقات الأحادية هى عندما تكون تعدديتها واحد - متعدد ومتعدد - متعدد. وفيما يلى شرح مفصل لعملية تحويل كل منهما للنموذج العلاقى.

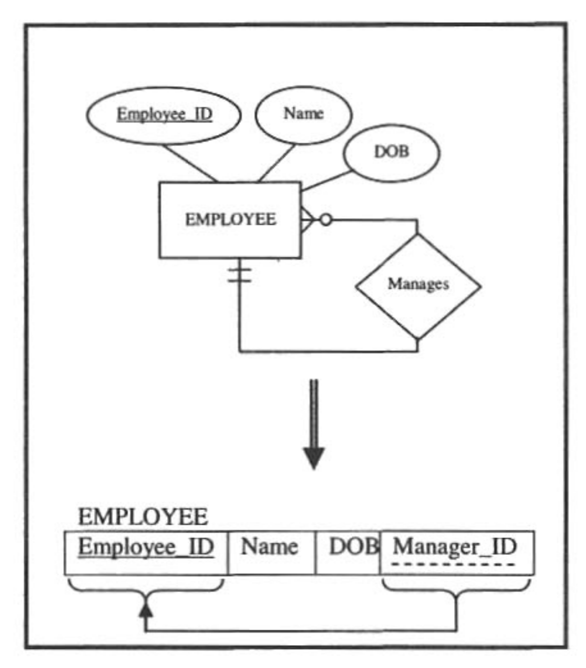
٥-١-٥-١ التعامل مع العلاقات الأحادية ذات التعددية واحد - متعدد:

يتم تحويـل العلاقة الأحادية ذات التعددية واحد - متعدد بخطوتين: فى الخطوة الأولـى يتم تعريف جدول للكينونة التى تربط بين حالاتها العلاقة الأحادية كما سـبق أن أوضحنـا فى قاعدة التحويل رقـم (١) أعلاه. أما فى الخطوة الثانية فيتم إضافة حقل إضافى لجدول الكينونة الذى تم تعريفه، بحيث يكون هذا الحقل مفتاحاً خارجياً يشـير إلى الجدول نفسه، وبحيث تكون نوعية بياناته ومداها من نوعية بيانات ومدى المفتاح الرئيسى لجدول الكينونة الذى تم تعريفه.

ويمثل الشـكل رقم (٥-١١) علاقة أحادية ذات تعددية واحد - متعدد وهى علاقة «يديـر» (Manages) التى تربط بين حـالات الكينونة «موظـف» (Employee). فالمدير الواحـد فى المنظمة يدير صفـراً أو أكثر من الموظفين. أما الموظف الواحد فيجب أن يرأسـه (أو يديره) مدير واحد فقط(*). وحسـب قاعدة التحويـل أعلاه، يتم تعريف جدول لكينونة «الموظف» بحيث يحتوى على الخصائص المرتبطة بالكينونة وهى: رقم الموظف، واسـم الموظف، وتاريخ ميلاده. كما يتم تعريف المفتاح الرئيسى للجدول وهو رقـم الموظف. بعد ذلك تتم إضافة حقل جديد للجدول وهو حقل «رقم المدير» وذلك لتمثيل علاقة «يدير». كما يتم تعريف الحقل الجديد بنفس نوع ومدى المفتاح الرئيسى لجدول كينونة الموظف، وعلى أسـاس أنه مفتاح خارجى يشـير إلـى جدول الكينونة نفسـه. وبهذه الطريقة يمكـن التعرف على مدير كل موظف فـى المنظمة من خلال المفتاح الخارجى المدون فى سجل الموظف.

(*) قد لا يكون لمدير المنظمة من يرأسه، وفى هذه الحالة يتم إدخال قيمة مساوية لرقم الموظف فى حقل المفتاح الخارجى للدلالة على أن الموظف يدير نفسه، مع ضرورة تعطيل العمل فى القيود فى أثناء إدخال سجل لمثل هذا الموظف، كما سنوضح فى الفصل السابع عند شرح طريقة تعطيل العمل بالقيود. وبديلاً لذلك يمكن تعريف العلاقة بأنها اختيارية عوضاً عن كونها إجبارية (واحد).

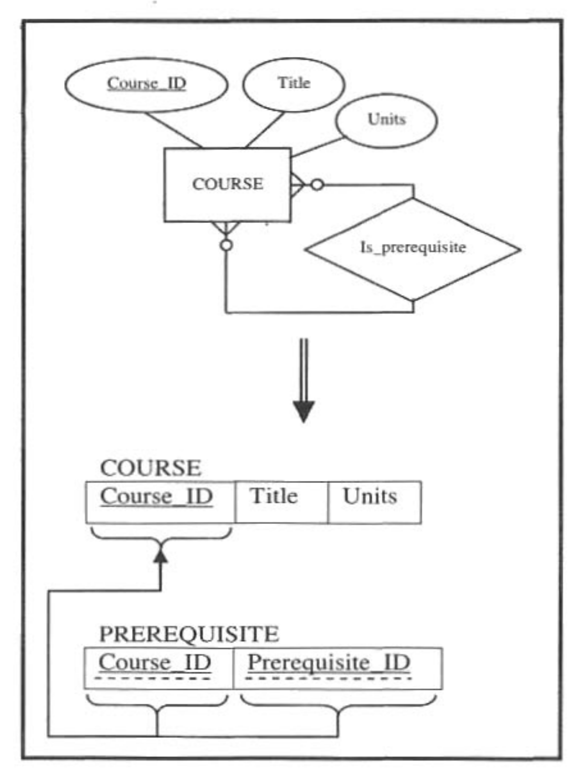
شكل رقم (٥-١١): تحويل العلاقة الأحادية ذات التعددية واحد - متعدد إلى النموذج العلاقى



٥-١-٥-٢ التعامل مع العلاقات الأحادية ذات التعددية متعدد - متعدد:

عنــد وجود علاقــة أحادية ذات تعددية متعدد - متعدد فإنــه يتم تعريف جدولين فــى أثناء عملية التحويل للنموذج العلاقــى. الجدول الأول يمثل الكينونة التى ترتبط بالعلاقة والجدول الثانى يمثل العلاقة نفســها. ويكون المفتاح الرئيسى للجدول الذى يمثل العلاقة عبارة عن مفتاح مركب يعرف فيه المفتاح الرئيسى لجدول الكينونة مرتين، كما يعرف كل جزء منه على أنه مفتاح خارجى يشير إلى جدول الكينونة. ويمكن تشبيه عمليــة التحويل هذه بعملية تحويل العلاقة الثنائية ذات التعددية متعدد - متعدد التى يتم فيها تعريف ثلاثة جداول بحيث يكون أحد هذه الجداول ممثلاً للعلاقة التى تربط بين الكينونتين المرتبطتين بها، ويتكون مفتاحها الرئيســى من حقول المفاتيح الرئيسية لكلتا الكينونتين التى تربط بينهما. ونظراً لوجود كينونة واحدة فى أية علاقة أحادية، فإن المفتاح الرئيســى لجدول العلاقة هو تكرار المفتاح الرئيسى لجدول الكينونة التى ترتبــط بالعلاقة. وفى حال ارتبطــت العلاقة بخصائص فإنه يتم تعريف حقل مقابل لكل خاصية ضمن جدول العلاقة.

شكل رقم (٥-١٢): تحويل العلاقة الأحادية ذات التعددية متعدد - متعدد إلى النموذج العلاقى



ويوضح الشكل رقم (٥-١٢) عملية تحويل علاقة أحادية ذات تعددية متعدد - متعدد. ويوجد فى الشكل كينونة «المادة الدراسية» (COURSE) التى ترتبط حالاتها مع بعض من خلال علاقة «متطلب دراسى» (Is_prerequisite). وتعنى هذه العلاقة أن كل مادة دراسية لها صفر أو أكثر من المتطلبات الدراسية. كما أن المادة الدراسية قد تكون متطلباً دراسياً لصفر أو أكثر من المواد الدراسية. وحسب قاعدة التحويل أعلاه، يتم تعريف جدولين: أحدهما لتمثيل كينونة «المادة الدراسية» والآخر لتمثيل علاقة «متطلب دراسى». ونظراً لأن علاقة «متطلب دراسى» لا ترتبط بأية خصائص وأن المفتاح الرئيسى لجدول المادة الدراسية يتكون من حقل واحد فقط، فإن جدول العلاقة يحتوى على حقلين فقط. وكل حقل منهما هو المفتاح الرئيسى لجدول الكينونة، مع الملاحظة بأن تسمية حقل المفتاح الرئيسى لجدول الكينونة ليس من الضرورى أن يكون نفسه فى الجدول الآخر مادام من نوعية البيانات نفسها وله مدى القيم نفسه.

ويصبح المفتاح الرئيسى للعلاقة هو حقلى المفتاح الرئيسى لجدول الكينونة مدمجين مع بعضهما. كما يتم تعريف كل واحد منهما على أنه مفتاح خارجى يشير لجدول الكينونة، كما هو موضح فى الشكل رقم (٥-١٢).

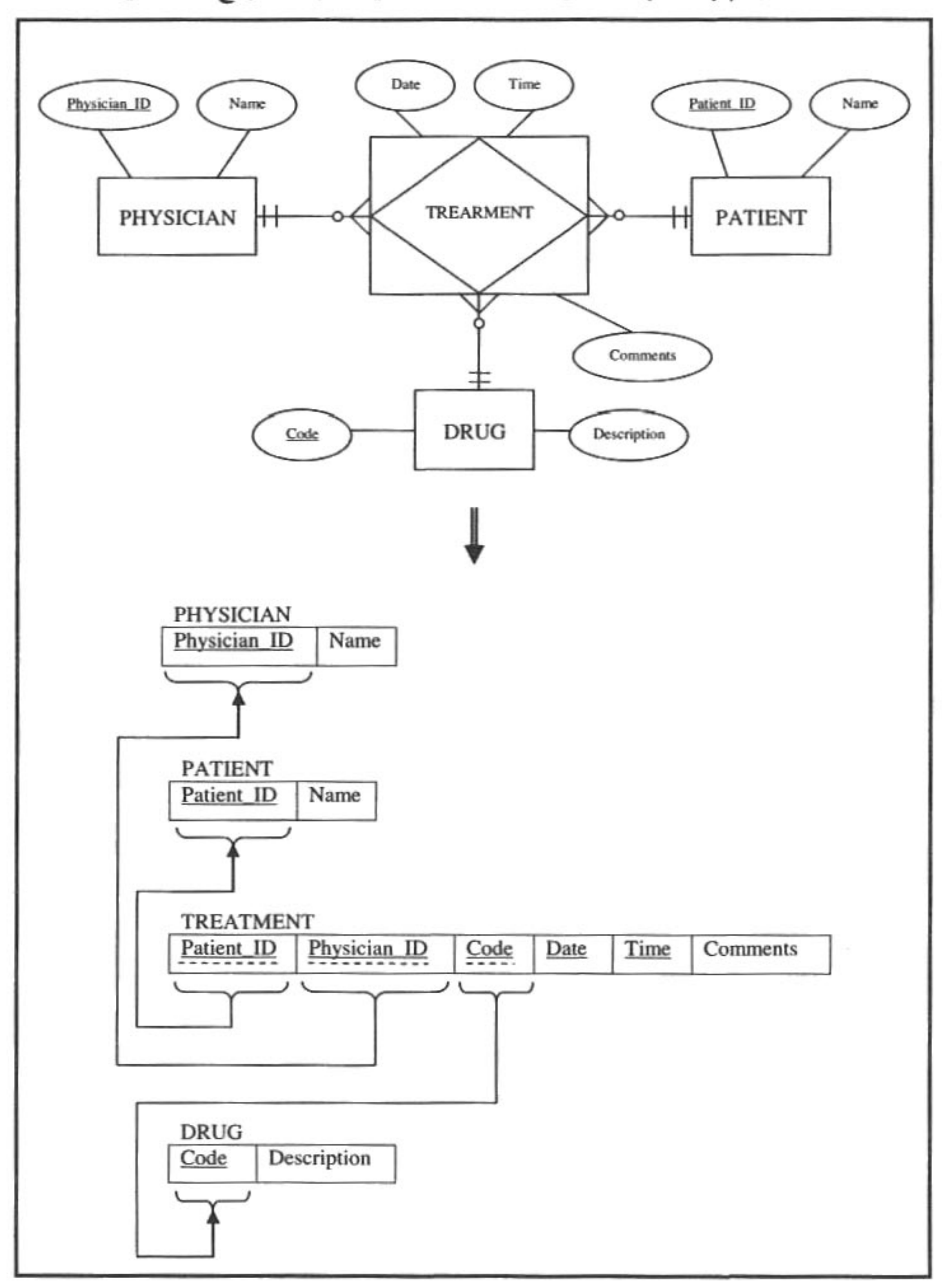
٥-١-٦ قاعدة التحويل السادسة: التعامل مع العلاقات الثلاثية (وما أعلى من ذلك):

العلاقة الثلاثية هى علاقة تربط بين ثلاثة أنواع من الكينونات. ويفضل تحويل العلاقة الثلاثية (والعلاقات ذات الدرجات الأعلى من ثلاثة) إلى علاقة مشاركة حتى يمكن توصيف قيود التعددية بشكل أدق. ولتحويل علاقة مشاركة تربط بين ثلاث كينونات (أو أكثر)، يتم إنشاء جدول لتمثيل العلاقة المشاركة (بالإضافة إلى جداول الكينونات التى تربط بينها العلاقة) بحيث يحتوى الجدول على حقول تمثل المفاتيح الرئيسية لجداول الكينونات التى تربط بينها العلاقة وحقول لتمثيل أية خصائص مرتبطة بالعلاقة نفسها. ويكون المفتاح الرئيسى لجدول العلاقة مكوناً من حقول المفاتيح الرئيسية لجداول الكينونات التى تربط بينها العلاقة. وفى بعض الأحيان يضاف لحقول المفاتيح الرئيسية لجداول الكينونات التى تربطها العلاقة حقول أخرى تمثل بعض خصائص العلاقة نفسها، وذلك عندما لا تمكننا المفاتيح الرئيسية لجداول الكينونات من التعرف على حالات العلاقة بشكل متفرد.

ويمثل الشكل رقم (٥-١٣) علاقة ثلاثية وهى علاقة «العلاج» (TREATMENT) التى تربط بين كينونة «الطبيب» (PHYSICIAN) وكينونة «المريض» (PATIENT) وكينونة «الدواء» (DRUG). ويمكن أن تقرأ هذه العلاقة على أن الطبيب يصف دواءً للمريض، وهو علاج المريض. وعليه فإن علاقة «العلاج» هى علاقة تربط بين ثلاث حالات فى الوقت نفسه: حالة من حالات الأطباء، وحالة من حالات المرضى، وحالة من حالات الدواء.

وحسب قاعدة التحويل أعلاه، يتم إنشاء جدول خاص بالعلاقة الثلاثية (أو العلاقة ذات الدرجة الأعلى من ثلاثة)، بالإضافة لجداول الكينونات التى تربط بينها العلاقة. ويحتوى جدول العلاقة على حقول تمثل المفاتيح الرئيسية لجداول الكينونات الثلاث وهى: «رقم الطبيب» (Physician_ID)، و«رقم المريض» (Patient_ID)، و«رمز الدواء» (Code). وتكون هذه الحقول الثلاثة جزءاً من المفتاح الرئيسى لجدول العلاقة بالإضافة إلى كونها مفاتيح خارجية تشير لجداول الكينونات الثلاث التى تربط بينها العلاقة. كما يتكون جدول العلاقة من حقول الخصائص المرتبطة بالعلاقة نفسها وهى: «التاريخ» (Date)، و«الوقت» (Time)، و«الملاحظات» (Comments).

شكل رقم (٥-١٣): تحويل العلاقة الثلاثية إلى النموذج العلاقى



ونظراً لأنه من الممكن أن يقوم المريض الواحد بمقابلة الطبيب نفسه وأخذ الدواء نفسه أكثر من مرة، فإن المفاتيح الرئيسية لجداول الكينونات الثلاث فقط لا تصلح لأن تكون مفتاحاً رئيسياً لجدول العلاقة، وذلك لأنها لا تمكن من التمييز بين حالات العلاقة بشكل منفرد. لذلك تم استخدام خاصية التاريخ وخاصية الوقت المرتبطتين بالعلاقة ليكونا جزءاً من المفتاح الرئيسى لجدول العلاقة. وبهذه الطريقة يمكن التعرف على حالات العلاقة بشكل منفرد؛ إذ إن المريض قد يقوم بمقابلة الطبيب نفسه وأخذ الدواء نفسه فى اليوم نفسه ولكن فى أوقات مختلفة. أما إذا افترضنا أنه ليس من الممكن أن يقوم المريض بمقابلة الطبيب نفسه وأخذ الدواء نفسه فى اليوم نفسه، فإنه يمكن الاكتفاء بخاصية التاريخ لتصبح جزءاً من المفتاح الرئيسى دون استخدام خاصية الوقت.

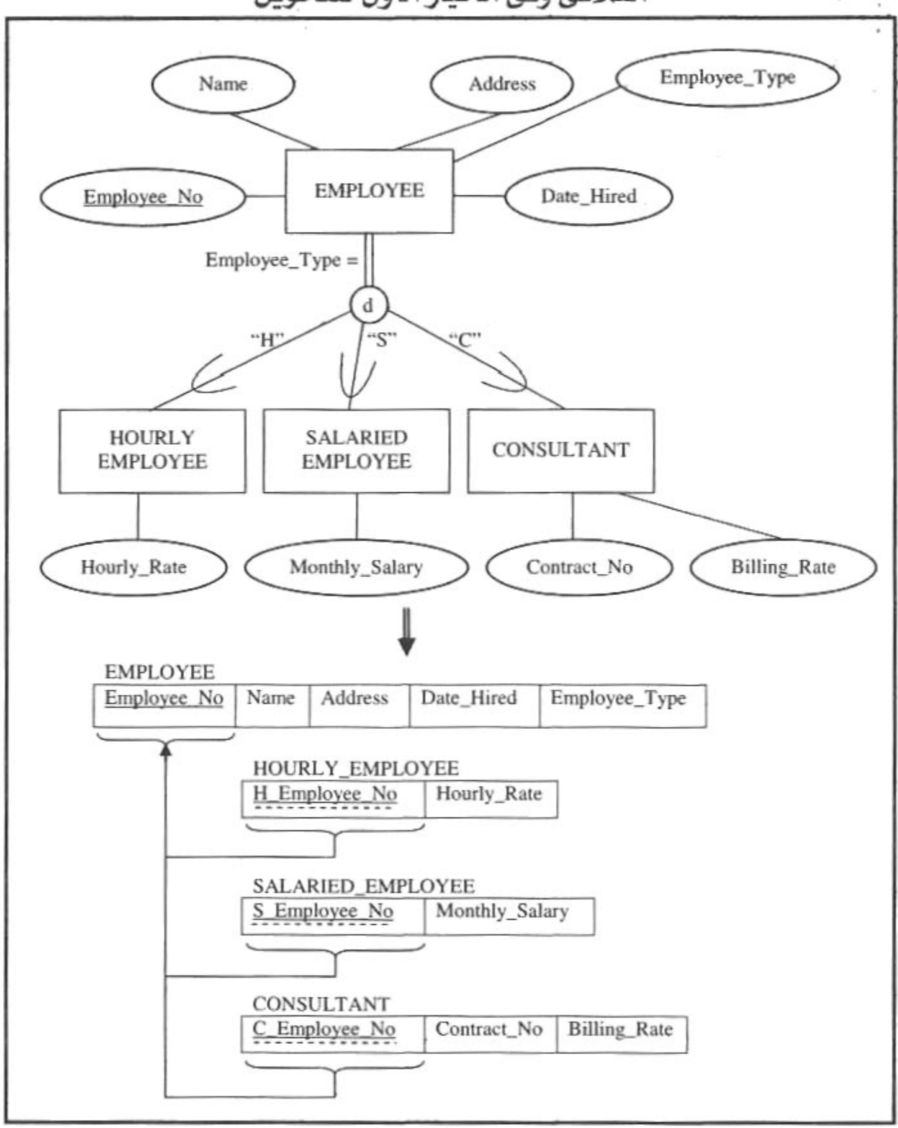
٥-١-٧ قاعدة التحويل السابعة: التعامل مع علاقات الأنواع الرئيسية والأنواع الفرعية:

لا يُمكِّن النموذج العلاقى حالياً من تمثيل علاقات الأنواع الرئيسية والأنواع الفرعية بشكل مباشر، إلا أنه يتوافر عدد من الخيارات لمصممى قواعد البيانات تمكنهم من تحويل علاقات الأنواع الرئيسية والأنواع الفرعية من النموذج المفاهيمى كينونة - علاقة إلى النموذج العلاقى. وفيما يلى شرح للخيارات الأربعة التى تعد الأكثر شيوعاً فى عملية التحويل (Elmasri and Navathe, 2004):

٥-١-٧-١ الخيار الأول:

يتم إنشاء عدد من الجداول بحيث يخصص واحد منها لتمثيل النوع الرئيسى، وواحد لكل نوع من أنواعه الفرعية. ويتكون جدول النوع الرئيسى من عدد من الحقول يكون مكافئاً لعدد الخصائص المرتبطة به فى نموذج كينونة - علاقة، ويكون المفتاح الرئيسى لجدول النوع الرئيسى هو الخاصية (أو مجموعة الخصائص) المعرفة للنوع الرئيسى فى نموذج كينونة - علاقة. كما تتم إضافة حقل أو أكثر فى جدول النوع الرئيسى لتمثيل «مميز الأنواع الفرعية». ويتم إنشاء جدول لكل نوع فرعى يرتبط بالنوع الرئيسى فى نموذج كينونة - علاقة بحيث يكون عدد حقول الجدول المنشأ لنوع فرعى معين مكافئاً لعدد خصائص النوع الفرعى فى نموذج كينونة - علاقة، بالإضافة إلى حقل (أو أكثر) لتمثيل المفتاح الرئيسى لجدول النوع الرئيسى. ويعنى هذا أن كل جدول لنوع فرعى يجب أن يحتوى على المفتاح الرئيسى لجدول النوع الرئيسى. كما يتم تعريف حقل المفتاح الرئيسى للنوع الرئيسى الذى تم إنشاؤه فى جدول النوع الفرعى على أنه مفتاح رئيسى للنوع الفرعى، وفى الوقت نفسه مفتاح خارجى يشير إلى جدول النوع الرئيسى.

شكل رقم (٥-١٤): تحويل علاقة الأنواع الرئيسية والأنواع الفرعية إلى النموذج العلاقى وفق الخيار الأول للتحويل



ويمثل الشكل رقم (٥-١٤) نوعاً رئيسياً وهو «الموظف» (EMPLOYEE) الذى يرتبط به ثلاثة أنواع فرعية من الموظفين وهى: «موظفو أجر الساعات» (HOURLY_EMPLOYEE)، و«موظفو الأجر الشهرى» (SALARIED_EMPLOYEE)، و«المستشارون»

(CONSULTANTS). وحسب القاعدة أعلاه، يتم إنشاء أربعة جداول، أحدها لتمثيل النوع الرئيسى والثلاثة المتبقية لتمثيل الأنواع الفرعية. كما يتم إدراج الخصائص المشتركة للأنواع الفرعية (وهى تلك المرتبطة بالنوع الرئيسي) كحقول ضمن جدول النوع الرئيسى بما فيها الخاصية المعرفة، وهى خاصية «رقم الموظف» (Employee_No). يتم أيضاً إضافة حقل للخاصية التى تميز بين الأنواع الفرعية وهى خاصية «نوع الموظف» (Employee_Type). أما بالنسبة لجداول الأنواع الفرعية فيتكون كل واحد منها من حقول تمثل الخصائص التى يتفرد بها عن بقية الأنواع الفرعية، بالإضافة إلى حقل المفتاح الرئيسى لجدول النوع الرئيسى الذى يعرف ضمن جدول النوع الفرعى على أساس أنه مفتاح رئيسى، وفى الوقت نفسه مفتاح خارجى يشير إلى جدول النوع الرئيسى.

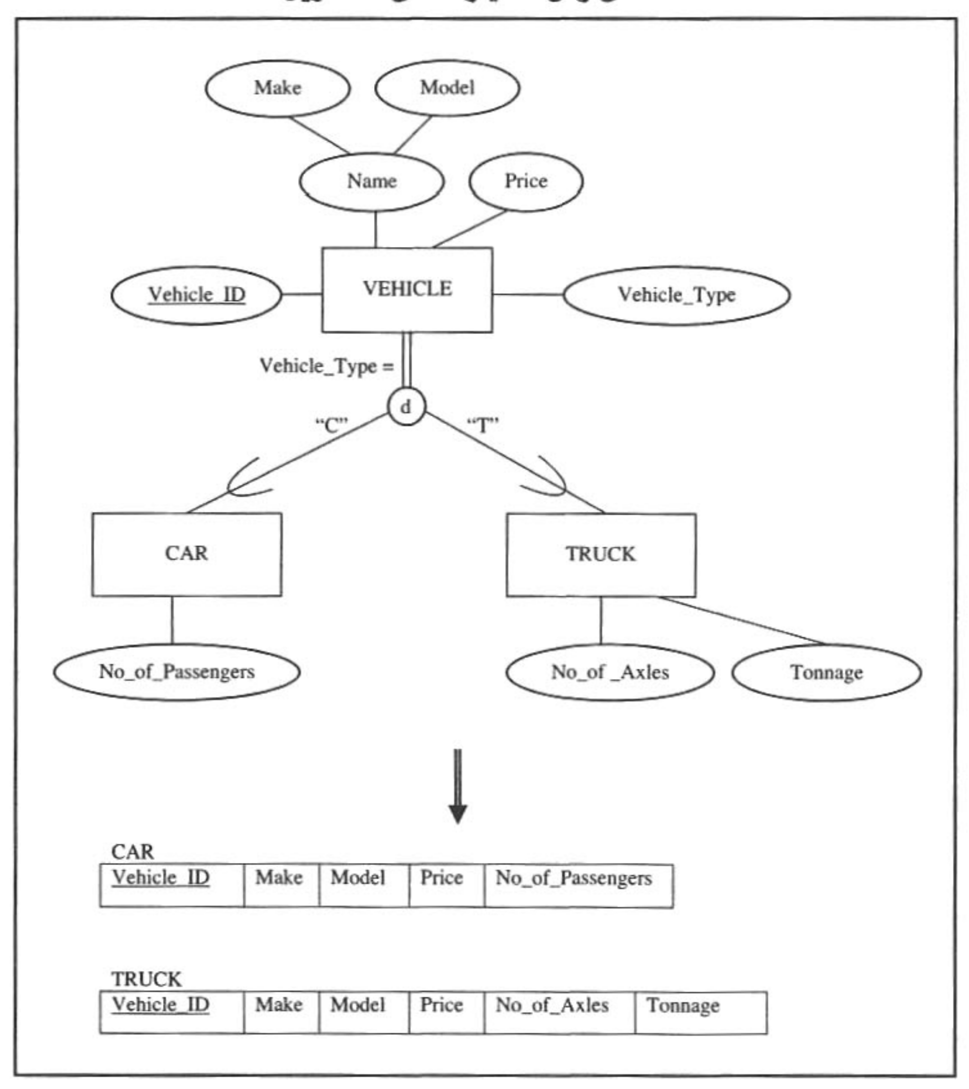
٥-١-٧-٢ الخيار الثانى:

يتم إنشاء عدد من الجداول بحيث يكون كل واحد منها ممثلاً لنوع فرعى واحد دون تمثيل النوع الرئيسى. ويحتوى جدول كل نوع فرعى على حقول لتمثيل الخصائص التى ينفرد فيها النوع الفرعى، بالإضافة إلى الخصائص المشتركة بين الأنواع الفرعية وهى تلك المرتبطة بالنوع الرئيسى. ويكون المفتاح الرئيسى لجدول أى نوع فرعى هو الحقل الذى يمثل الخاصية المعرفة التى ترتبط بكينونة النوع الرئيسى. كما يستغنى عن تمثيل خاصية «مميز الأنواع الفرعية» باستخدام طريقة التحويل هذه. ويمكن استخدام طريقة التحويل هذه عندما يكون قيد التخصيص كاملاً بمعنى أن أية حالة من حالات النوع الرئيسى لا بد أن توجد ضمن أحد أنواعه الفرعية، ويكون قيد الانفصال كاملاً أيضاً، بمعنى أنه لا يمكن أن توجد حالة ما ضمن أكثر من نوع فرعى واحد فى وقت واحد.

ويوضح الشكل رقم (٥-١٥) كينونة النوع الرئيسى، وهو «المركبة» التى يرتبط فيها نوعان فرعيان هما «السيارة» و «الشاحنة». ونظراً لأن كل مركبة لا بد أن تمثل ضمن أحد الأنواع الفرعية، فإن قيد التخصيص هو تخصيص كامل، كما هو موضح فى الشكل بالخطين المزدوجين اللذين يصلان النوع الرئيسى بنقطة التفرع. أما قيد الانفصال فهو انفصال كامل؛ وذلك لأن السيارة لا يمكن أن تكون شاحنة أو بالعكس. فى مثل هذه الحالة يمكن استخدام الطريقة أعلاه فى عملية تحويل النموذج المفاهيمى إلى النموذج العلاقى حيث يتم إنشاء جدولين أحدهما لتمثيل النوع الفرعى «سيارة»، والثانى لتمثيل النوع الفرعى «شاحنة». كما يتم إدراج حقول لتمثيل الخصائص المشتركة (التى ترتبط بالنوع الرئيسي) ضمن جدولى كلا النوعين

الفرعيـــين بالإضافة للخصائص المميـــزة لكل نوع منهما ضمن الجـــدول الممثل للنوع الفرعى. كما تعرف خاصية المعرف المرتبطة بالنوع الرئيسى، فى كلا الجدولين، على أنها المفتاح الرئيسى لهما.

شكل رقم (٥-١٥): تحويل علاقة الأنواع الرئيسية والأنواع الفرعية إلى النموذج العلاقى وفق الخيار الثانى للتحويل



CAR

Vehicle_ID	Make	Model	Price	No_of_Passengers

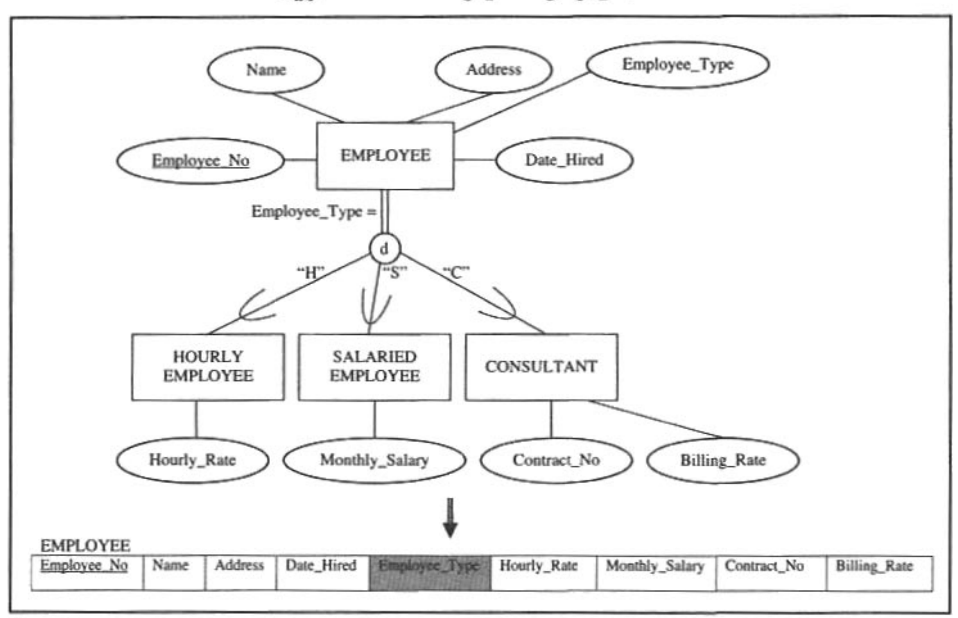
TRUCK

Vehicle_ID	Make	Model	Price	No_of_Axles	Tonnage

٥-١-٧-٣ الخيار الثالث:

يتم إنشــاء جدول واحــد فقط يتكون من حقول تمثل جميع الخصائص المشــتركة للأنواع الفرعية بالإضافة إلى حقول تمثل الخصائص المميزة لكل نوع فرعى. كما يتم إضافة حقل لتمثيل خاصية «مميز الأنواع الفرعية». ويمكن استخدام طريقة التحويل هــذه عندمــا يكون قيد الانفصال كاملاً، بمعنى أنه لا يمكــن أن توجد حالة ما ضمن أكثر من نوع فرعى واحد فى الوقت نفســه، بغض النظر عن قيد التخصيص، ســواء كان كاملاً أو جزئياً.

شكل رقم (٥-١٦): تحويل علاقة الأنواع الرئيسية والأنواع الفرعية إلى النموذج العلاقى وفق الخيار الثالث للتحويل



ويمثل الشكل رقم (٥-١٦) نوعاً رئيسياً وهو «الموظف» (EMPLOYEE) الذى يرتبط بــه ثلاثة أنواع فرعية مــن الموظفين، وهــى: «موظفو أجر الســاعات» (HOURLY_EMPLOYEE)، و«موظفو الأجر الشهرى» (SALARIED_EMPLOYEE)، و«المستشارون» (CONSULTANTS). ونظــراً لكون قيد الانفصال هو انفصال كامل، فإنه يمكن تطبيق القاعدة أعلاه بحيث يتم إنشــاء جــدول واحد فقط لتمثيل النوع الرئيســى والأنواع

الفرعية الثلاثة. ويتكون الجدول من حقول تمثل الخصائص المشتركة للأنواع الفرعية (وهى تلك المرتبطة بالنوع الرئيسي) بالإضافة إلى حقول تمثل الخصائص التى يتميز بهـا كل نوع فرعى. كما يحتوى الجدول على حقل يمثـل مميز الأنواع الفرعية، وهو حقــل «نوع الموظف» (Emplyee_Type). وعنــد إضافة موظف إلى الجدول، يتم إدخال كافة بيانات الموظف ونوعه. أما بالنســبة للحقول التى لا تنطبق على الموظف فتدخل فيها القيمة غير المعرفة (NULL).

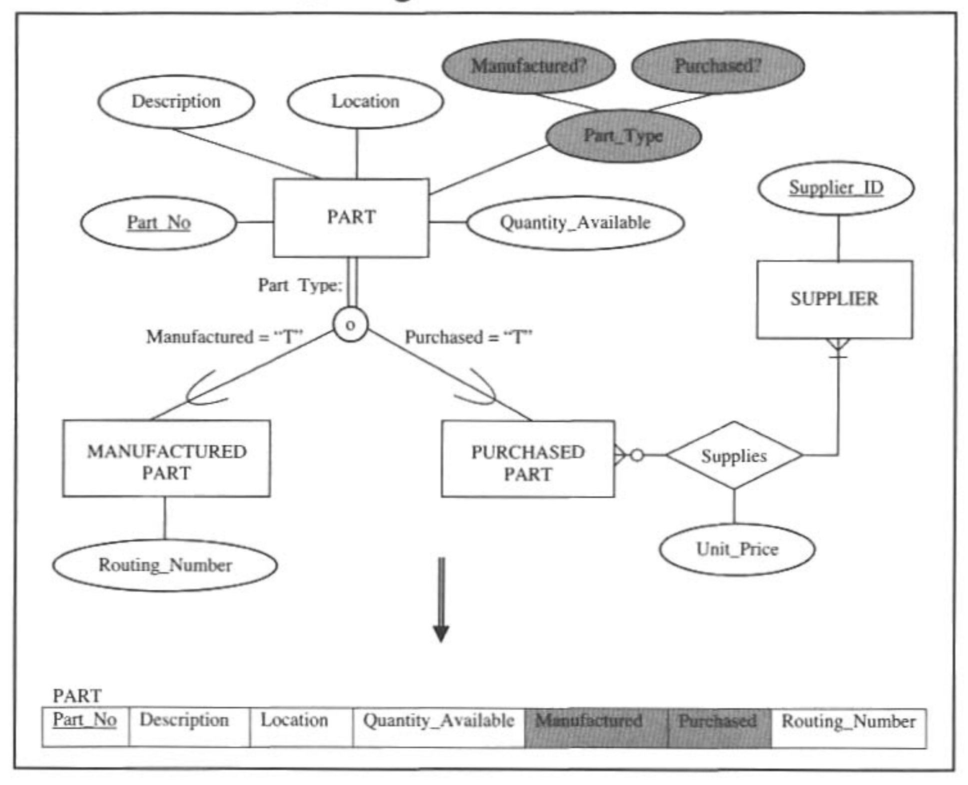
٥-١-٧-٤ الخيار الرابع:

يتم إنشــاء جدول واحــد فقط يتكون من حقول تمثل جميع الخصائص المشــتركة للأنــواع الفرعية، بالإضافة إلى حقول تمثــل الخصائص المميزة لكل نوع فرعى. كما يتم إضافة عدد من الحقول يســاوى عدد الأنواع الفرعية، بحيث يقابل كل حقل منها نوعــاً فرعياً واحداً. وتعرف هذه الحقول الإضافية على أنها ذات بيانات ثنائية القيم (Boolean Data Type). وتكون قيمة أى حقل من هذه الحقول الإضافية مساوية للقيمة «صح» (True) إذا كانت الحالة المدخلة تتبع للنوع الفرعى المقابل للحقل الإضافى ثنائى القيــم. أمــا إذا لم تكن الحالة المدخلة تابعة لذات النــوع الفرعى، تكون قيمة الحقل الإضافى المقابل للنوع الفرعى هى «خطأ» (False). ويمكن اســتخدام طريقة التحويل هذه عندمــا يكون قيد الانفصال جزئياً بمعنى أنه من الممكن أن توجد حالة ما ضمن أكثر من نوع فرعى واحد فى الوقت نفسه، بغض النظر عن قيد التخصيص، سواء كان كاملاً أم جزئياً. كما يمكن اســتخدام هذه الطريقة أيضاً عندما يكون قيد الانفصال كاملاً.

ويمثل الشــكل رقم (٥-١٧) نوعاً رئيســياً وهو «قطعة غيار» (PART) الذى يرتبط بــه نوعان فرعيان: النوع الأول منهما يمثل قطــع الغيار المصنعة داخلياً (فى المنظمة نفسها) بمسمى (MANUFACTURED_PART)، والنوع الثانى يمثل قطع الغيار المشتراة (PURCHASED_PART). ونظراً لكون قيد الانفصال هو انفصال متداخل، إذ إن بعض قطع الغيار قد تكون مصنعة داخلياً وفى الوقت نفســه مشــتراة؛ يمكن حينئذٍ تطبيق القاعــدة أعلاه بحيث يتم إنشــاء جدول واحــد فقط لتمثيل النوع الرئيســى ونوعيه الفرعين. ويتكون الجدول من حقول تمثل الخصائص المشتركة للأنواع الفرعية (وهى تلك المرتبطة بالنوع الرئيســى) بالإضافة إلى حقــول تمثل الخصائص التى يتميز بها كل نــوع فرعى. كما يحتوى الجدول على حقلــين ذَوَىْ نوعية بيانات ثنائية القيم هما

حقــل «مصنعة» (Manufactured) وحقل «مشــتراة» (Purchased). وتكون قيمة أى حقل من هذين الحقلين إما «صح» وإما «خطأ». فعندما تكون قطعة الغيار مشــتراة فقط تكون قيمة حقل «مشتراة» صح، وتكون قيمة حقل «مصنعة» خطأ. أما إذا كانت قطعة الغيــار مصنعة داخلياً فقط فتكون قيمة حقل «مصنعة» صح، وقيمة حقل «مشــتراة» خطأ. وفى حال كانت بعض من قطعة الغيار مصنعة داخلياً وبعض منها مشــتراة فإن قيمة كلا الحقلين تكون صح.

شكل رقم (٥-١٧): تحويل علاقة الأنواع الرئيسية والأنواع الفرعية إلى النموذج العلاقى وفق الخيار الرابع للتحويل



Part_No	Description	Location	Quantity_Available	Manufactured	Purchased	Routing_Number

٥-١-٧-٥ فوارق خيارات تصميم علاقات الأنواع الرئيسية والأنواع الفرعية:

يعد الخيار الأول والخيار الثانى من خيارات التصميم التى ينتج عنها أكثر من جدول، فى حين الخيار الثالث والخيار الرابع يُعدَّان من خيارات التصميم التى ينتج عنها جدول واحد فقط. كما يعتبر الخيار الأول لعملية تحويل علاقات الأنواع الرئيسية والأنواع الفرعية خياراً عاماً، بمعنى أنه يمكن استخدامه بغض النظر عن القيود المفروضة على النوع الرئيسى وأنواعه الفرعية (وهما قيد التخصيص وقيد الانفصال). إلا أن هذا الخيار يتطلب عملية «ربط تساوى» (Equi-Join) على المفتاح الرئيسى بين جدول النوع الرئيسى وجدول أى نوع فرعى للحصول على جميع بيانات النوع الفرعى. أما إذا أردنا الحصول على بيانات جميع الأنواع الفرعية فإن هذا يتطلب عملية «اتحاد خارجى» (Outer Union) بعد عملية «ربط تساوى» بين الأنواع الفرعية بالنوع الرئيسى.

أما الخيار الثانى فلا يتطلب عملية «ربط تساوى» للحصول على جميع بيانات نوع فرعى معين؛ لأن هذه البيانات متوافرة بالكامل ضمن جدول النوع الفرعى نفسه. إلا أن هذا الخيار يستخدم فى حالة كون قيد التخصيص كاملاً، وقيد الانفصال كاملاً أيضاً. فإذا لم يكن قيد التخصيص كاملاً فإنه سيتم فقد بيانات الأنواع التى لم يتم تخصيصها. أما إذا كان قيد الانفصال متداخلاً فإنه سيتم تكرار تخزين بعض البيانات ضمن جداول الأنواع الفرعية. وكما هو الحال فى الخيار الأول، تستخدم عملية «الاتحاد الخارجى» (Outer Union) إذا أردنا الحصول على جميع الأنواع الفرعية، وذلك لأنه لا يحتوى أى من جداول الأنواع الفرعية على بيانات الأنواع الفرعية كافة.

يستخدم الخيار الثالث عندما يكون قيد الانفصال كاملاً. ويتم استخدام أحد حقول الجدول باعتباره مميزاً للنوع الفرعى بحيث تحدد القيمة المخزنة فى هذا الحقل لكل حالة مخزنة فى الجدول النوع الفرعى الذى تتبعه الحالة. وعندما يكون قيد التخصيص كاملاً، فإنه لا بد أن تتبع كل حالة مخزنة فى الجدول لأحد الأنواع الفرعية، مما يعنى وجود قيمة فى حقل مميز الأنواع الفرعية تحدد النوع الفرعى الذى تتبعه الحالة. أما إذا كان قيد التخصيص جزئياً فإن هذا يعنى إمكانية وجود حالات ضمن الجدول لا تتبع لأى نوع فرعى. وفى هذه الحالة تترك قيمة حقل مميز الأنواع الفرعية غير معرفة (NULL) كما تترك جميع الحقول التابعة للأنواع الفرعية غير معرفة أيضاً.

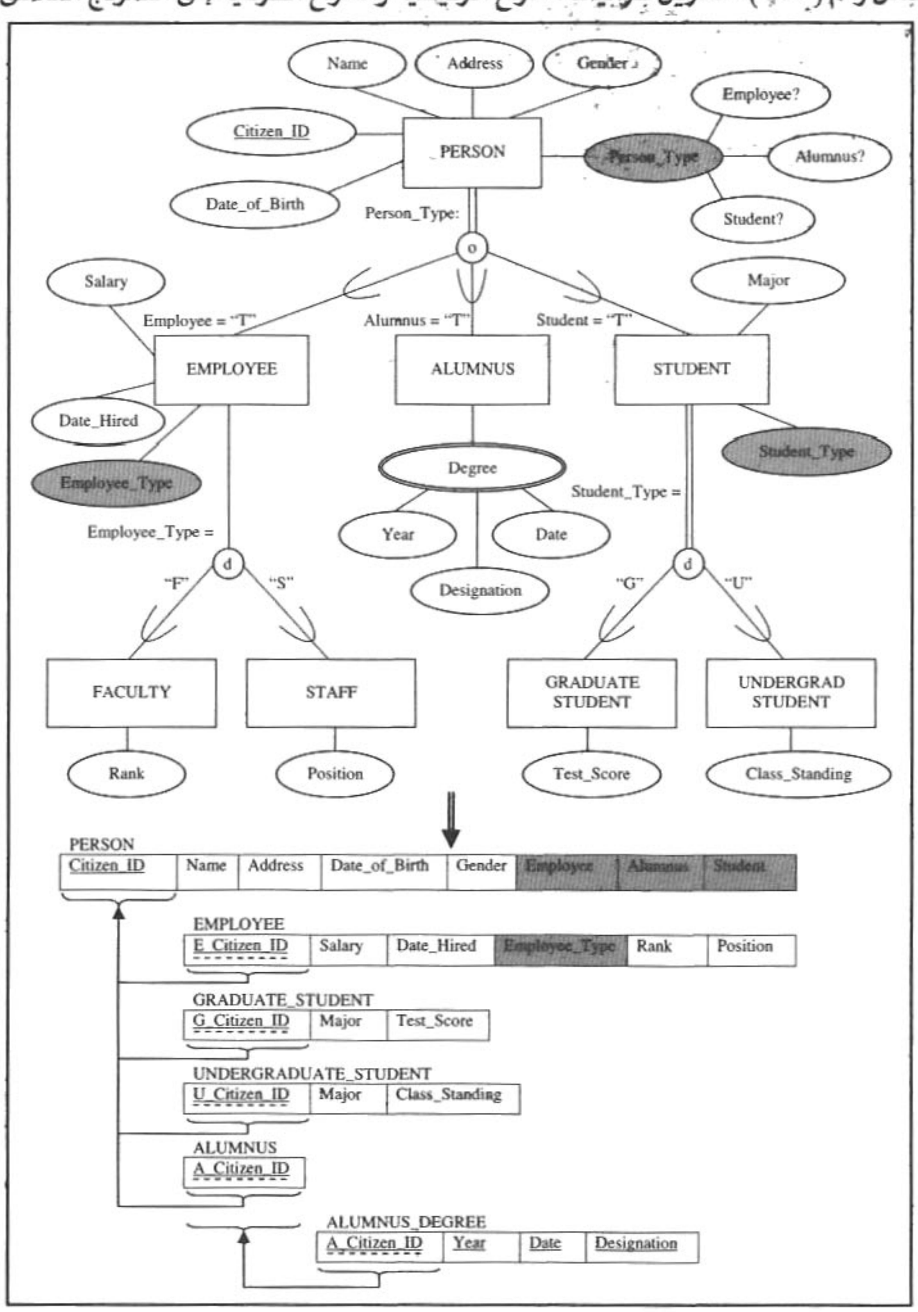
صُمم الخيار الرابع بحيث يمكن من تمثيل علاقة الأنواع الرئيسية والأنواع الفرعية عندما يكون قيد الانفصال متداخلاً. وباستخدام هذا الخيار يتم تعريف عدد إضافى من الحقول فى الجدول الذى يمثل النوع الرئيسى والأنواع الفرعية، بحيث يكون عدد الحقول الإضافية هذه مساوياً لعدد الأنواع الفرعية. وتعرف هذه الحقول الإضافية على أنها ذات بيانات ثنائية القيم (Boolean Data Type). وتكون قيم أى حقل من هذه الحقول الإضافية مساوية للقيمة «صح» (True) إذا كانت الحالة المدخلة تتبع للنوع الفرعى المقابل للحقل الإضافى ثنائى القيم. أما إذا لم تكن الحالة المدخلة تابعة للنوع الفرعى نفسه، تكون قيمة الحقل الإضافى المقابل للنوع الفرعى هى «خطأ» (False). كما تكون قيم جميع الحقول التابعة للنوع الفرعى، فى هذه الحالة، غير معرَّفة (NULL). وكما هو الحال فى طريقة التحويل الثالثة، فإن طريقة التحويل هذه تعفينا من إجراء أية عملية ربط أو اتحاد للحصول على كل بيانات الأنواع الفرعية.

إن الخيارات الأربعة أعلاه تعطى مصممى قواعد البيانات المرونة الكافية لتحديد الطريقة المناسبة فى تحويل علاقات الأنواع الرئيسية والأنواع الفرعية. فالخيار الأول والخيار الثانى ينتج عنهما أكثر من جدول، مما يستدعى إجراء عمليات ربط بين الجدول واتحاد فيما بينها، مما يؤدى إلى استغراق وقت أطول فى تنفيذ الاستفسارات مقارنة بالطريقة الثالثة والطريقة الرابعة. إلا أن الطريقة الأولى والطريقة الثانية لا تستنزفان المساحة التخزينية، وذلك لعدم وجود الكثير من القيم غير المعرَّفة ضمن الجداول مقارنة بالطريقة الثالثة والطريقة الرابعة. على النقيض من ذلك فإن الطريقة الثالثة والطريقة الرابعة أسرع فى تنفيذ الاستفسارات من الطريقة الأولى والطريقة الثانية، وذلك لكون جميع بيانات الأنواع الفرعية متوافرة فى جدول واحد، مما يعنى عدم الحاجة لإجراء أية عملية ربط أو اتحاد، إلا أن هاتين الطريقتين تستنزفان الكثير من المساحة التخزينية، وخاصة عندما تكون الحقول المرتبطة بكل نوع فرعى كثيرة نسبياً مما ينتج عنه الكثير من القيم غير المعرَّفة ضمن بيانات الجدول.

٥-١-٧-٦ تحويل هرميات الأنواع الرئيسية والأنواع الفرعية:

عندما نقوم بعملية تحويل هرميات من علاقات الأنواع الرئيسية والأنواع الفرعية فإنه ليس من الضرورى اتباع نفس خيار التحويل لجميع الأنواع الرئيسية والأنواع الفرعية، وإنما يمكن استخدام خيارات مختلفة. ويوضح الشكل رقم (٥-١٨) خيارات مختلفة لتحويل هرمية من الأنواع الرئيسية والأنواع الفرعية.

شكل رقم (٥-١٨): تحويل هرميات الأنواع الرئيسية والأنواع الفرعية إلى النموذج العلاقى

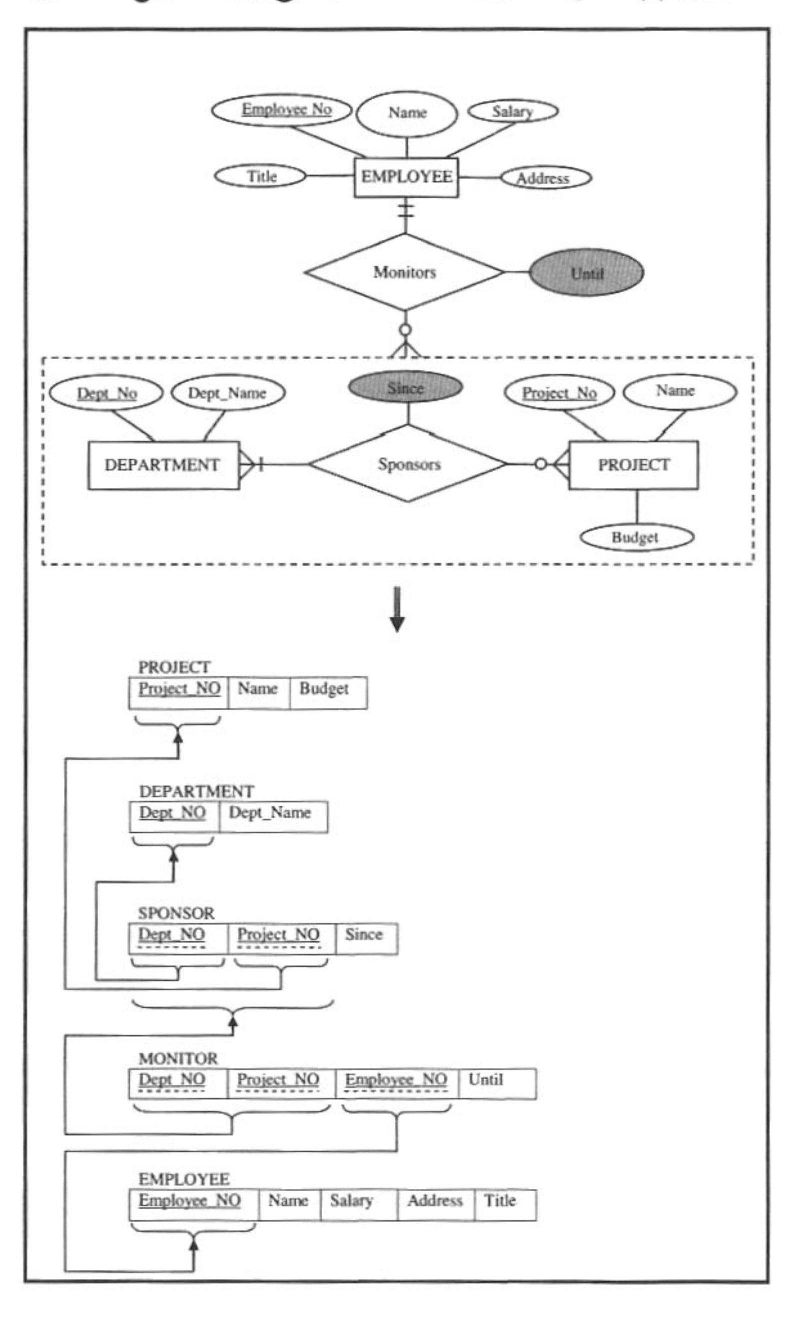


ويلاحظ فى عملية التحويل الممثلة فى الشكل أنه تم استخدام الخيار الأول لتمثيل علاقة النوع الرئيسى «شخص» (PERSON) وأنواعه الفرعية، حيث تم إدراج الخصائص المشتركة لجميع فئات الأشخاص ضمن جدول «شخص». ولكون قيد الانفصال انفصالاً متداخلاً، فقد تم إضافة ثلاثة حقول تبين نوعية الشخص فيما إذا كان موظفاً أو طالباً أو خريجاً أو أية توليفات أخرى، مثل أن يكون الشخص خريجاً من الجامعة، وفى الوقت نفسه موظفاً فيها. وعند تحويل النوع الفرعى «موظف» تم استخدام الخيار الثالث حيث تم إنشاء جدول واحد لجميع أنواع الموظفين مع إضافة مميز لنوع الموظف يبين إنْ كان الموظف عضواً لهيئة التدريس أو موظفاً غير ذلك (من العاملين فى إحدى الوظائف الإدارية). وعند تحويل النوع الفرعى «طالب» (STUDENT) تم استخدام الخيار الثانى حيث تم إنشاء جدولين: جدول لتمثيل طلبة درجة البكالوريوس، وجدول لتمثيل طلبة الدراسات العليا. أما فيما يتعلق بالنوع الفرعى «خريج» (ALUMNUS) فإنه لا يرتبط بأى أنواع فرعية ولكنه يرتبط بخاصية مركبة وهى «الدرجة العلمية» (Degree) التى تم تحويلها حسب قاعدة التحويل رقم (١).

٥-١-٨ قاعدة التحويل الثامنة: التعامل مع التجميع:

عند وجود تجميع، يتم تحويل الكينونات والعلاقات المجمعة حسب قواعد التحويل التى سبق شرحها أعلاه. أما بالنسبة لعلاقة التجميع التى تربط بين كينونة ما، من جانب، والتجميع، من جانب آخر، فيتم التعامل معها وكأنها علاقة تربط بين كينونتين. فعلى سبيل المثال، لنفترض وجود التجميع الممثل فى الشكل رقم (٥-١٩). إن هذا التجميع (الممثل داخل الشكل المستطيل ذى الخط المتقطع) يربط بين كينونة «القسم» وكينونة «المشروع» من خلال علاقة «الدعم المالى». ولتحويل هذا التجميع نستخدم قاعدة التحويل رقم (٥-١-٣-٢) التى توضح طريقة تحويل العلاقات الثنائية ذات التعددية متعدد - متعدد. وباستخدام هذه الطريقة، يتم إنشاء ثلاثة جداول: جدولين لتمثيل الكينونتين اللتين ترتبط بينهما العلاقة الثنائية «الدعم المالي» (SPONSOR)، والجدول الثالث لتمثيل العلاقة نفسها. وينتج عن هذه الخطوة ثلاثة جداول هى جدول «القسم» (DEPARTMENT)، وجدول «المشروع» (PROJECT)، وجدول «الدعم المالى» (SPONSOR). ويوضح الشكل المفاتيح الرئيسية والمفاتيح الخارجية لهذه الجداول.

شكل رقم (٥-١٩): تحويل علاقات التجميع إلى النموذج العلاقى



أمـــا عملية تحويل علاقة التجميع وهى «المتابعة» (Monitors) فهى شـــبيهة بتحويل العلاقات الثنائية التى تربط بين كينونتين، إذ يتم إنشاء جدول خاص بالعلاقة يحتوى على حقل لتمثيل المفتاح الرئيسى لجدول كينونة الموظفين وهو «رقم الموظف» (Employee_No)، وحقول المفتاح الرئيسى لجدول علاقة «الدعم المادى» وهى «رقم القسم» (Dept_No) و «رقم المشـــروع» (Project_No)، كما يحتـــوى على حقل لتمثيل الخاصية المرتبطة بالعلاقة، وهى «حتى» (Until). وتعرف الحقول الثلاثة للمفاتيح الرئيسية مجتمعة على أنها المفتاح الرئيســـى لجدول «المتابعة»، كما يعرف كل مفتاح على حدة على أنه مفتاح خارجى. فالمفتاح (Employee_No) يعد جزءاً من المفتاح الرئيسى لجدول العلاقة وفى الوقت نفســـه يعد مفتاحاً خارجياً يشـــير لجدول «الموظف». أما حقل «رقم القســـم» وحقل «رقم المشـــروع» فهما مجتمعين يعدان جزءاً من المفتاح الرئيســـى للعلاقة وفى الوقت نفسه يعدان مفتاحاً خارجياً يشير لجدول علاقة «الدعم المادى».

وهنالـــك بعض الحالات الخاصة التى تمكننا من تحســـين عملية التحويل، وذلك من خلال حذف الجدول الذى يربط بين الكينونتين المجتمعتين. ففى مثالنا الســـابق، يمكن حذف جدول «الدعم المادى» لو لم ترتبط علاقة «الدعم المادى» بخاصية خاصة فيها. إلا أنه بشكل عام لا يمكن التحسين على التصميم السابق ما لم يتحقق الشرطان التاليان:

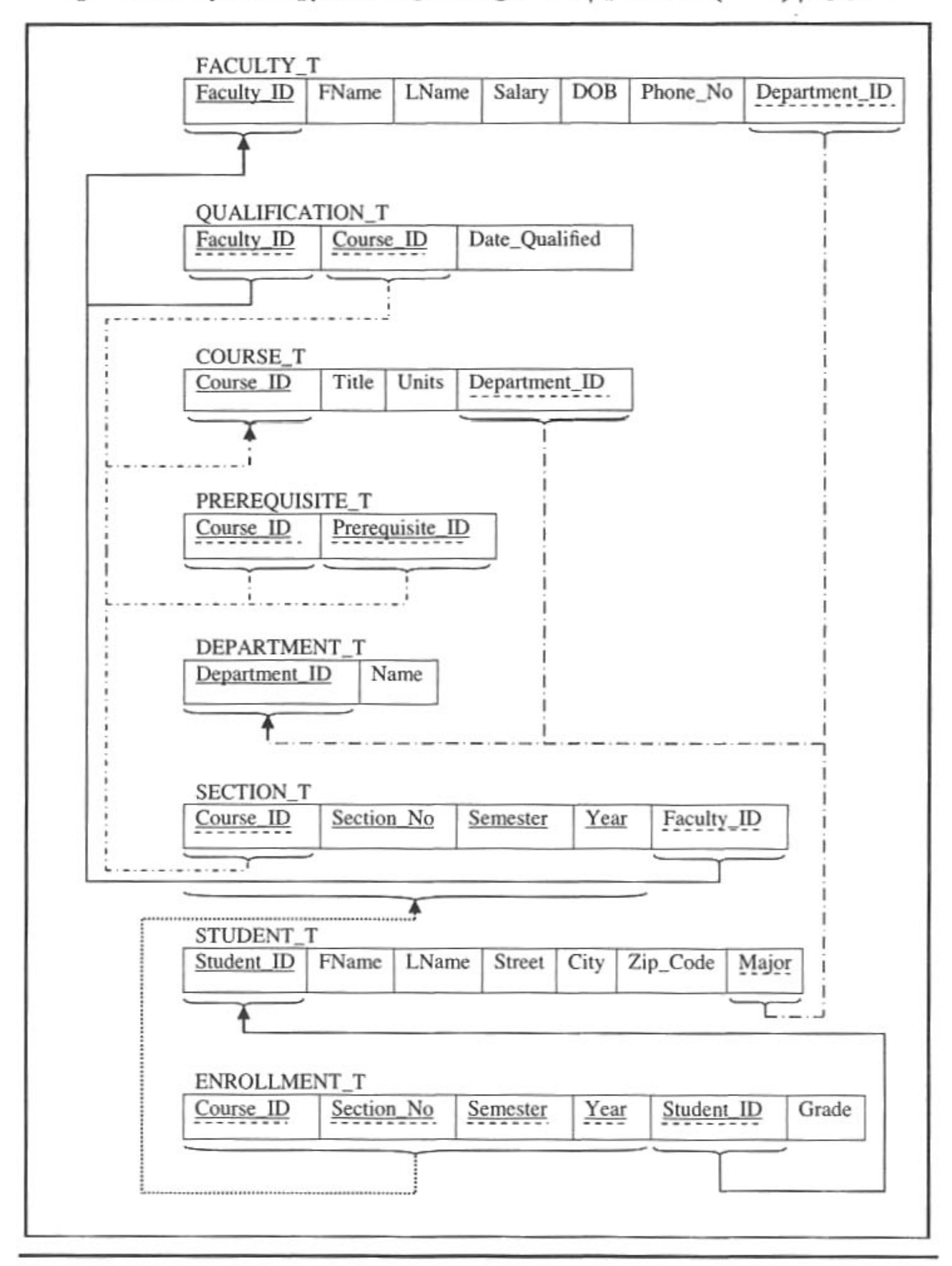
١- أن لا ترتبط العلاقة التى بين الكينونات المجمعة بخصائص خاصة فيها.
٢- أن تكون كل حالة مجمعة مرتبطة بعلاقة التجميع.

ففى مثالنا السابق، كل «دعم مالي» يجب أن يرتبط بموظف واحد على الأقل. وبناء على ذلك، فإن هذا الشرط الثانى منطبق على مثالنا، ولكن الشرط الأول غير منطبق. لذا، فإننا لا نستطيع إلغاء جدول «الدعم المالى» من تصميم الجداول العلاقية.

٥-٢ التصميم المنطقى للحالة الدراسية:

على الرغم من أن أدوات هندســـة البرمجيات تقوم بتحويل النموذج المفاهيمى إلى النموذج العلاقى بشـــكل تلقائـــى، إلا أنه من الأهمية التعرف علـــى خطوات التحويل هذه. لذا فإن هذا الفصل قد ركز على قواعد تحويل النموذج المفاهيمى إلى النموذج العلاقى فى خطوة تدعى «التصميم المنطقى» لقواعد البيانات. وهذه الخطوة تترجم تصميم قاعدة البيانات من نموذج عالى المستوى، قريب من مستوى إدراك المستفيدين من قاعدة البيانات لبياناتهم، إلى تصميم لقاعدة البيانات نفســـها ولكن للنموذج الذى سيتم بناء قاعدة البيانات عليه.

شكل رقم (٥-٢٠): التصميم المنطقى الكامل لقاعدة بيانات الجامعة الأهلية



وبنـــاء علـــى خطوات التحويل التى تم شـــرحها فى هذا الفصل، فإن الشـــكل رقم (٥-٢٠) يوضـــح التصميم المنطقى الكامل لقاعـــدة بيانات الجامعة الأهلية. ويلاحظ فى الشـــكل إضافة الحرف "T" بعد اســـم كل جدول، وذلك للتفريق بين جداول قاعدة البيانات وبقيـــة أنواع هياكل قاعدة البيانات مثل الفهارس والمنظورات - التى ســـيتم التطرق إليها فى الفصول المتعلقة بلغة الاستفســـار البنائية (الفصل الســـابع والفصل الثامن). ويلاحظ فى الشـــكل أيضاً اســـتخدام خطوط مختلفة بعضها ما هو متصل والبعض الآخر منقط، إلا أن هذا الاختلاف فى طبيعة الخطوط لا يدل على اختلاف فـــى المعنى المقصـــود بها؛ إذ إنها جميعاً تســـتهدف ربط المفاتيـــح الخارجية بالمفاتيح الرئيســـية التى تشير إليها، وأن هذا الاختلاف فى طبيعة الخطوط جاء بشكل متعمد حتى تسهل عملية تتبع الخطوط فى الشكل فقط.

الفصل السادس

تطبيع العلاقات والتصميم المادى لقواعد البيانات العلاقية

سبق أن أشرنا فى الفصل السابق أن مرحلة التصميم المنطقى لقواعد البيانات تتكون من خطوتين رئيسيتين: فى الخطوة الأولى يتم تحويل النموذج المفاهيمى إلى نموذج قاعدة البيانات المستخدمة، وهو النموذج العلاقى الذى يمثل أحد محاور هذا الكتاب. وقد تم شرح هذه الخطوة فى الفصل السابق. أما فى الخطوة الثانية فيتم تحسين تصميم قاعدة البيانات الناتجة من عملية التحويل بحيث تحتوى على أقل قدر ممكن من البيانات المتكررة حتى يتم تجنب المشكلات التى قد تنتج عن عمليات التعديل على محتويات قاعدة البيانات. وتدعى هذه الخطوة بعملية «التطبيع» (Normalization)، التى تمثل موضوع الجزء الأول من هذا الفصل.

أما الجزء الثانى من هذا الفصل فيركز على مرحلة التصميم المادى لنظم قواعد البيانات الذى يهدف إلى إنشاء تصميم يمكّن من تخزين البيانات بشكل يوفر الأداء المناسب لنظام إدارة قاعدة البيانات على اختلاف حجم العمليات التى تنفذ عليها. ويعنى هذا، وعلى خلاف التصميم المفاهيمى والتصميم المنطقى، أن التصميم المادى يوضح الكيفية التى ستخزن وتعالج فيها البيانات، لا على الكيفية التى يتم من خلالها التعرف على البيانات والعلاقات فيما بينها أو طريقة تمثيلها وفق النموذج العلاقى أو نماذج البيانات الأخرى.

٦-١ التطبيع (Normalization):

عند شرح النموذج المفاهيمى، أَعملنا الحَدْس فى أثناء عملية التعرف على الكينونات وتجميع الخصائص التابعة لكل منها. وبعد ذلك تم استخدام خطوات محددة لتحويل النموذج المفاهيمى إلى علاقات. إلا أن الاستناد إلى الحدس فقط فى تصميم قواعد البيانات غير كافٍ ولا يمكننا من قياس أو معرفة جودة الجداول المكونة لقاعدة البيانات. لذلك فإننا بحاجة إلى طريقة رسمية واضحة المعالم والأسس النظرية التى تمكننا من معرفة جودة الجداول التى تم تصميمها. وهذه الطريقة الرسمية

تســـمى «التطبيع» (Normalization). ولكن قبـــل البدء فى التعرف على مفهوم التطبيع والخطوات التى تتبع للتأكد من جودة جداول قاعدة البيانات، سنقوم بإيضاح المقصود بالجداول جيدة البناء (Well-Structured Relations).

٦-١-١ الجداول جيدة البناء (Well-Structured Relations):

مـــن المنطقى أن يحتـــوى أى جدول على أقـــل قدر ممكن مـــن التكرارية؛ وذلك لأن تكرارية البيانات، كما ســـبق أن أوضحنا فى الفصل الأول، تؤدى إلى مشـــكلات أو عدم تناســـق فى البيانات ما لم يتم التعرف على مكامنها بشـــكل دقيق والتحكم فيها بشـــكل كامل. لذا فإن أى جدول يجب أن يحتوى على أقل قدر ممكن من التكرارية فى بياناته، بحيث يمكّن المستفيدين من التعامل مع محتوياته، من خلال عمليات الحذف والتحديث والإضافة، دون حدوث مشكلات أو عدم تناسق فى البيانات. ويمثل الجدول رقم (٦-١) جـــدولاً جيد البناء؛ لأن كل صف فيه يمثل البيانـــات المتعلقة بعضو هيئة تدريس واحد ودون وجود أية تكرارية فى بيانات عضو هيئة التدريس. كما أن بإمكان المستفيدين من الجدول حذف أى ســـجل فيه أو تحديث أى حقل من حقوله أو إضافة أى ســـجل جديد دون أية مشكلات (أو عدم تناسق) فى بياناته، وذلك لأن أياً من هذه التغييرات محصورة فى سجل واحد من سجلات الجدول.

جدول رقم (٦-١): مثال لجدول جيد البناء

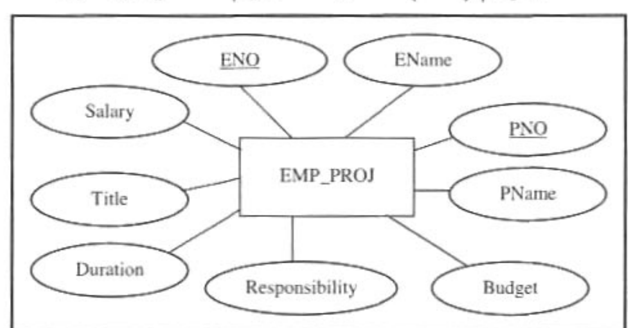
FACULTY

Faculty_ID	FName	LName	Phone_NO	Salary	DOB
200	Khalid	Aloufi	454-2341	35000	22/05/1963
220	Fahad	Alhamid	456-7733	25900	07/10/1970
310	Saleh	Aleesa	454-8932	30000	13/09/1966
320	Mohammed	Alhamad	454-5412	44000	13/05/1965
330	Ghanim	Alghanim	456-2234	44500	12/08/1969
340	Ibraheem	Alsaleh	454-1234	25000	20/01/1970
400	Ahmad	Alotaibi	454-4563	33900	17/05/1971

على النقيض من الجدول الســـابق فـــإن التصميم الممثل بالنمـــوذج المفاهيمى فى الشـــكل رقم (٦-١) يعد تصميماً ســـيئاً حيث ســـينتج عنه جدول ســـيئ البناء أيضاً. والسبب وراء ذلك أن الجدول رقم (٦-٢) الناتج عن هذا التصميم يحتوى على الكثير

من التكرارية فى بياناته. فعلى سبيل المثال، تتكرر البيانات الخاصة باسم الموظف ومسمى الوظيفة والراتب لكل من الموظف رقم "E2" والموظف رقم "E3" فى صفين من صفوف الجدول. ونتيجة لذلك فإننا لو حاولنا تعديل راتب أو رقم هاتف أى من هذين الموظفين فإنه يجب علينا إجراء التحديث فى سجلين من سجلات الجدول عوضاً عن سجل واحد (كما هو الحال فى الجدول رقم (٦-١)). ونتيجة لهذه التكرارية فى بيانات الجدول، فإنه من الممكن أن ينتج عن عمليات التعديل عليه مشكلات (أو عدم تناسق) فى البيانات. ويوجد هناك ثلاثة أنواع من مشكلات التعديل (Modification Anomalies) هى: مشكلة الإضافة (Insertion Anomaly)، ومشكلة الحذف (Deletion Anomaly)، ومشكلة التحديث (Update Anomaly)، وهى كما يلى:

شكل رقم (٦-١): مثال لتصميم مفاهيمى سيئ



١- **مشكلة الإضافة:** لو أردنا إضافة سجل لموظف جديد فإننا يجب أن نضيف قيمة لحقل «رقم المشروع» (PNO) بالإضافة إلى بيانات الحقول المتعلقة بالموظف؛ وذلك لأن حقل رقم المشروع يعد جزءاً من المفتاح الرئيسى للجدول، ولا يمكن أن تكون قيمته غير معرفة. لذلك فإن هذا الجدول يحتوى على مشكلة، وإن هذه المشكلة تتسبب فى عدم إمكانية إضافة سجلات جديدة للموظفين إلا بإضافة بيانات تتعلق بالمشاريع.

٢- **مشكلة الحذف:** لو قمنا بحذف سجل الموظف رقم "E1" فإننا لن نحذف البيانات المتعلقة بهذا الموظف فحسب، ولكنه سيتم حذف البيانات المتعلقة بالمشروع رقم "P1" كذلك. لذلك فإن هذا الجدول يحتوى على مشكلة، وإن هذه المشكلة تتسبب فى عدم إمكانية حذفنا لبيانات الموظفين دون حذف بيانات تتعلق بالمشاريع.

٣- **مشكلة التحديث:** لو أردنا تغيير أرقام هواتف أو رواتب أى من الموظفين رقم "E2" ورقم "E3" فإنه يجب علينا إجراء مثل عمليات التحديث هذه فى أكثر من سجل، وإلا أصبحت حقول بيانات الجدول غير متناسقة فى محتوياتها.

جدول رقم (٦-٢): مثال لجدول سيئ البناء ناتج عن تصميم مفاهيمى سيئ

EMP_PROJ

ENO	EName	Title	Salary	PNO	PName	Budget	Duration	Responsibility
E1	Saleh Aloufi	Electrical Eng	40000	P1	Database System	170000	12	Manager
E2	Ahmad Alhamid	System Analyst	35000	P3	Human Resources	190000	20	System Analyst
E2	Ahmad Alhamid	System Analyst	35000	P2	Inventory	220000	10	System Analyst
E3	Mohamed Alhamad	Mechanical Eng.	37000	P3	Human Resources	190000	10	Consultant
E3	Mohamed Alhamad	Mechanical Eng.	37000	P4	Maintenance	230000	14	Consultant
E4	Khalid Alsaleh	Programmer	29000	P2	Inventory	220000	26	Programmer
E5	Ibraheem Alotaibi	Database Admin.	45000	P2	Inventory	220000	14	Manager
E6	Mishal Aleesa	Programmer	29000	P4	Maintenance	230000	16	Programmer
E7	Abdullah Alghanim	Programmer	29000	P3	Human Resources	190000	12	Manager
E8	Turky Alsalman	Secretary	25000	P3	Human Resources	190000	12	Project Secretary

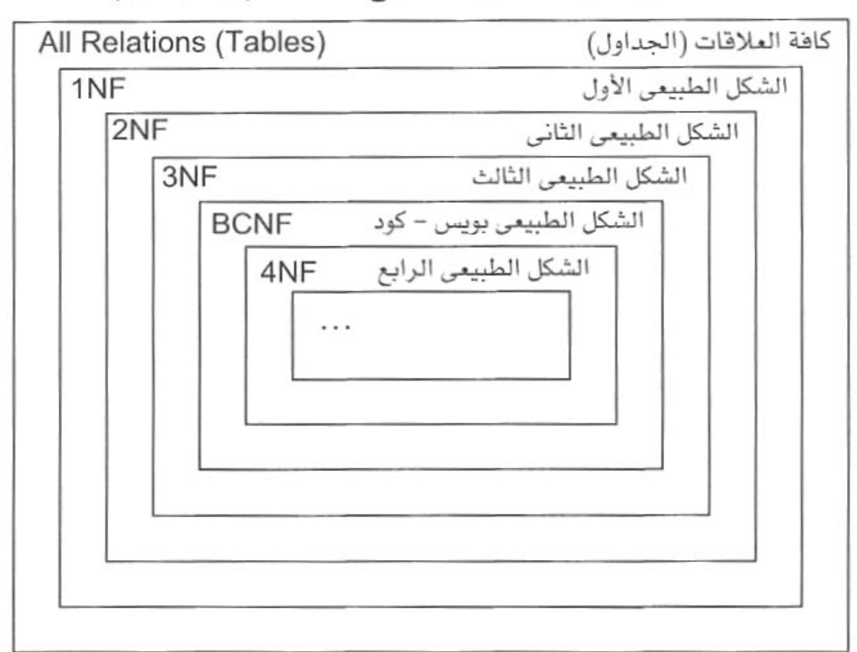
وتدل المشكلات السابقة فى الجدول على أنه سيئ التصميم، وذلك لكونه يحتوى على بيانات تتعلق بشيئين مختلفين أحدهما هو «الموظف»، والثانى هو «المشروع». وللتعرف على وجود مثل هذه المشكلات فى أى جدول ومعالجتها فإننا نستخدم تطبيع الجدول.

٦-١-٢ مستويات التطبيع:

إن عملية تطبيع العلاقات (أو الجداول) «عملية رسمية» (Formal Process) تمكننا من التعرف على مكامن المشكلات فى الجداول التى تم تصميمها فى أثناء الخطوة الأولى من عملية التصميم المنطقى، وذلك قبل الانتقال إلى مرحلة التصميم المادى الذى نقوم من خلالها ببناء قاعدة البيانات. لذا فإن التطبيع يعد أداة لتحسين تصميم الجداول الناتجة من الخطوة الأولى للتصميم المنطقى بحيث تتحقق عليها بعض الشروط التى تمنع من التكرارية غير المرغوب فيها للبيانات. وفى أثناء عملية

التطبيـــع يتم اختبار كل جدول للتأكد من تحقيقه لشـــروط أحد الأشـــكال الطبيعية. ويتم فى أثناء عملية التطبيع النظر فى تصميم كل جدول وتجزئته إلى أكثر من جدول بغية تحســـين تصميم الجدول الأساسى ليتوافق مع الخصائص المطلوب أن يتحلى بها الجدول. لذا فإن عملية التطبيع تعد عملية هرمية من الأعلى إلى الأسفل تهدف إلى الفصل بين المفاهيم (أو الأشـــياء) التى نقوم بنمذجتها. ويوضح الشـــكل رقم (٦-٢) مســـتويات الأشـــكال الطبيعية، بحيث أنه كلما زاد رقم الشـــكل الطبيعى (وصولاً إلى الداخل)، كانت الشروط المصاحبة للشكل الطبيعى أكثر شدة من الشكل الطبيعى الذى يسبقه، وبحيث يقلل من تكرارية البيانات التى يقبلها الشكل الطبيعى الذى قبله.

شكل رقم (٦-٢): مستويات تطبيع العلاقات (أو الجداول)



٦-١-٣ الاعتماديات الوظيفية ((Functional Dependencies (FDs)):

تعتمـــد عملية تطبيع الجداول على ما يعرف بالاعتماديـــات الوظيفية (Functional Dependencies). والاعتماديـــة الوظيفية هى قيد بين حقلين أو مجموعتين من الحقول فى الجدول بحيث إن أحد الحقلين أو إحدى المجموعتين تحدد وبشـــكل منفرد الحقل أو المجموعـــة الأخرى من الحقول. وفـــى أية حالة من الحالات التـــى قد يكون عليها

الجدول. ويعنى هذا أن الحقل الواحد قد يعتمد وظيفياً على حقلين أو أكثر من حقول الجـــدول. ففى جدول «الموظف – المشـــروع» (EMP_PROJ) أعلاه، يعتمد كل من حقل «المدة» (Duration) وحقل «المسئولية» (Responsibility) وظيفياً على حقلى «رقم الموظف» (ENO) وحقل «رقم المشـــروع» (PNO) مدمجين مع بعضهما. ويعنى هذا أن قيمة حقل «رقم الموظف» وقيمة حقل «رقم المشـــروع» مجتمعين يحددان قيمة كل من حقل «المدة» وقيمة حقل «المســئولية» بشكل منفرد فى جميع ســجلات الجدول سواء تلك المدونة فيــه فعلياً أو تلك التى قد تدون فيه مســتقبلاً. ويتــم تمثيل مثل هاتين الاعتماديتين الوظيفيتين كما يلى:

(ENO, PNO) ⟶ Duration
(ENO, PNO) ⟶ Responsibility

وتعنى الاعتمادية الوظيفية الأولى أنه يمكن معرفة (أو تحديد) الفترة الزمنية التى عمل فيها أى موظف على أى مشـــروع، وفى أية حالة يكون عليها محتوى الجدول، من خلال معرفة رقم الموظف ورقم المشروع. أما الاعتمادية الثانية فتعنى أن مسئولية أى موظف فى أى مشـــروع يمكن معرفتها من خلال رقم الموظف ورقم المشروع. ويلاحظ هنـــا أنه لا يمكن تحديد «المدة» أو «المســئولية» من خـــلال معرفة رقم الموظف أو رقم المشـــروع فحســـب، ولكنه يجب معرفة الاثنين معاً لتحديد كل من «المدة» و«المسئولية» بشكل منفرد. كما يمكن تمثيل الاعتماديتين الوظيفيتين أعلاه كما يلى:

(ENO, PNO) ⟶ (Duration, Responsibility)

ويعنى التمثيل أعلاه، أن الحقول الواقعة فى الجهة اليســرى من الســهم، وتدعى المحددات (Determinants)، تحدد، وبشـــكل منفرد، الحقـــول الواقعة فى الجهة اليمنى من السهم. ففى التمثيل السابق، يحدد الحقلان «رقم الموظف» و«رقم المشروع»، معاً، كلاً من حقل «المدة» وحقل «المســئولية». ومن أمثلة الاعتماديات الوظيفية الأخرى فى الجدول ما يلى:

١– (ENO, PNO) → (EName, Title, Salary, PName, Budget, Duration, Responsibility):
يحدد رقم الموظف ورقم المشـــروع مجتمعين بقية حقـــول الجدول (وذلك لكونهما المفتاح الرئيسى للجدول).

٢- ENO → (EName, Title, Salary): يحدد رقم الموظف كلاً من اسم الموظف، ومسمى وظيفته، وراتبه.

٣- PNO → (PName, Budget): يحدد رقم المشروع اسم المشروع، وميزانية المشروع.

٤- Title → Salary: يحدد المسمى الوظيفى للموظف الراتب الذى يتقاضاه الموظف.

ونظراً لأن من خصائص المفتاح الرئيسى لأى جدول تحديد سجلات الجدول بشكل منفرد، فإن حقلى «رقم الموظف» و«رقم المشروع» يحددان كل حقل من حقول الجدول بشكل منفرد. فعلى سبيل المثال، إذا عرفنا أن رقم الموظف هو "E2" وأن رقم المشروع هو "P1" فإن هاتين القيمتين تحددان، وبشكل منفرد، بقية حقول السجل. وتوضح الاعتمادية الوظيفية الأولى أعلاه هذا المفهوم. أما الاعتمادية الوظيفية الثانية والثالثة فيوضحان أنه من خلال معرفة قيمة حقل «رقم الموظف» نستطيع معرفة بقية بيانات الموظف، ومن خلال معرفة قيمة حقل «رقم المشروع» نستطيع معرفة بيانات المشروع. وتسمى مثل هاتين الاعتماديتين اعتماديات وظيفية جزئية (Partial Functional Dependencies): لأن الحقول الواقعة فى الجهة اليمنى فى كل من الاعتماديتين الوظيفيتين تعتمد على جزء من حقول المفتاح الرئيسى وليس جميع حقوله. أما الاعتمادية الوظيفية الرابعة فتوضح أن معرفة قيمة حقل «مسمى الوظيفة» للموظف تمكننا من معرفة راتبه.

ويمكن تعريف الاعتمادية الوظيفية بشكل رسمى كما يلى:

إذا افترضنا وجود جدول اسمه "R" يحتوى على عدد من الحقول "A" بحيث إن $A = (A_1, A_2, ..., A_n)$ وكانت كل من "X" و "Y" تمثل مجموعة جزئية من حقول الجدول ($X \subseteq A,\ Y \subseteq A$)، وإذا كان لأى زوجين من السجلات فى الجدول، وليكونا t_1 و t_2، فى أية حالة صحيحة من حالات الجدول:

$$t_1[X] = t_2[X] \Rightarrow t_1[Y] = t_2[Y]$$

فإنه يوجد اعتمادية وظيفية فى الجدول "R" بين الحقل أو مجموعة الحقول الممثلة فى "X" و "Y" كما يلى:

$$X \rightarrow Y$$

وتعد الاعتمادية الوظيفية الرابعة أعلاه مثالاً جيداً لفهم تعريف الاعتماديات الوظيفية إذ إن تَسَاوى مسمى الوظيفة لأى اثنين من الموظفين يعنى بالضرورة تساوى المرتبات التى يتقاضاها كلا الموظفين. ويعنى أن مسمى الوظيفة يدل دائماً على الراتب الذى يتقاضاه الموظف، وكذلك هو الحال بالنسبة للاعتمادية الوظيفية الثانية والثالثة، فالاعتمادية الثانية تعنى أن أى سجلين يحتويان على «رقم الموظف» نفسه ستكون قيم كل من حقل «اسم الموظف» وحقل «مسماه الوظيفى» وحقل «راتبه» متساوية فيهما. أما الاعتمادية الوظيفية الثالثة فتعنى أن أى سجلين يحتويان على «رقم المشروع» نفسه ستكون قيم كل من حقل «اسم المشروع» وحقل «ميزانية المشروع» متساوية فيهما.

وبناء على تعريف الاعتماديات الوظيفية يمكن تعريف المفتاح الخارق لأى جدول، الذى سبق تعريفه (فى الجزء ٤-١-١-٢ من الفصل الرابع) على أنه مجموعة من الحقول تمكن من التعرف على سجلات الجدول بشكل منفرد، بشكل رسمى كما يلى:

إذا افترضنا وجود جدول اسمه "R" يحتوى على عدد من الحقول "A" بحيث إن $A = (A_1, A_2, ..., A_n)$ وإن مجموعة جزئية من حقوله، ولتكن $(K \subseteq A)$، تمثل مفتاحاً خارقاً للجدول، فإنه يجب لأية زوجين من السجلات فى الجدول، وليكونا t_1 و t_2، فى أى حالة صحيحة من حالات الجدول - أن يتحقق الشرط التالى:

$$t_1 [K] = t_2 [K] \Rightarrow t_1 = t_2$$

ويعنى التعريف أعلاه أنه لا يمكن أن يكون فى أى جدول سجلان مختلفان لهما المفتاح الخارق نفسه؛ لأن تَسَاوى قيم المفتاح الخارق فى سجلين يعنى بالضرورة أنهما عبارة عن سجل واحد. أما المفتاح المرشح فيكون فى هذه الحالة عبارة عن مفتاح خارق، ولكنه لا يحتوى على مفتاح خارق آخر بمعنى أنه لا يمكن أن نقوم بحذف أى حقل من حقوله مع الاستمرار فى التعرف على سجلات الجدول بشكل منفرد، كما يوضح التعريف الرسمى التالى:

إذا افترضنا وجود جدول اسمه "R" يحتوى على عدد من الحقول "A" بحيث إن $A = (A_1, A_2, ..., A_n)$ وإن مجموعة جزئية من حقوله، ولتكن "C" $(C \subseteq A)$، تمثل مفتاحاً مرشحاً للجدول، فإنه يجب لأي زوجين من السجلات فى الجدول، وليكونا t_1 و t_2، فى أى حالة صحيحة من حالات الجدول - أن يتحقق الشرط التالى:

$$t_1 [C\text{-}A_i] = t_2 [C\text{-}A_i] \nRightarrow t_1 = t_2$$

ويعنى التعريف أعلاه أنه إذا تســاوى سجلان من سجلات أى جدول فى قيم بعض حقول المفتاح المرشح (بعدإزالة بعض منها) فإن هذه الحقول لا تعنى أن السجلين هما فى الواقع يمثلان السجل نفسه، كما هو الحال فى تعريف المفتاح الخارق أعلاه.

وتقســم الحقول فى أى جدول إلى نوعين: النوع الأول هو الحقول الأولية، والنوع الثانى هو الحقول غير الأولية، كما يلى:

- **الحقل الأوَّلي** هو حقل ينتمى لأحد المفاتيح المرشحة.
- **الحقل غير الأوَّلي** هو حقل لا ينتمى لأى مفتاح مرشح.

وبناء على التعاريف والمفاهيم الســابقة، نقدم فيما يلى الأشــكال الطبيعة الثلاثة الأولى التى قام «كود» باقتراحها (Codd, 1972) لتصبح سلســلة توصلنا إلى الخصائص المرغوب فيها فى هياكل الجداول التى يوفرها الشكل الطبيعى الثالث.

٦-١-٣-١ الشكل الطبيعى الأول (First Normal Form (1NF)):

يشــترط لأى جدول فى شــكله الطبيعى الأول أن يحتوى علــى قيمة واحدة فقط فــى أى حقل من حقوله ممــا يعنى أنه لا يمكن لأى جدول فى شــكله الطبيعى الأول أن يحتــوى على حقل متعدد القيم. ويعد الشــكل الطبيعــى الأول من ضمن التعريف الرسمى لهياكل الجداول العلاقية، حيث إن أى جدول علاقى لا يمكن أن يحتوى على حقول متعددة القيم. وقد تم تعريف هذا الشكل الطبيعى تاريخياً لتأكيد أن الجداول العلاقية يجب أن لا تحتوى على حقول متعددة القيم.

ولإيضاح طريقة تطبيع الجداول إلى الشــكل الطبيعى الأول لنفترض الجدول رقم (٦-٣) الذى يتكون من أربعة حقول هى: حقل «اســم القســم»، وحقل «رقم القسم»، وحقل «رقم الموظف» الذى يرأس القســم، وحقل «الموقع». ومن المفترض فى الجدول أنه يوجد لبعض الأقسام أكثر من موقع.

جدول رقم (٦-٣): جدول ليس فى الشكل الطبيعى الأول

DEPARTMENT

DNO	DName	D_MGR_NO	Location
10	Research	10101010	(Riyadh)
20	Computer Center	20202020	(Riyadh, Jeddah, Dammam)
30	Administration	30303030	(Riyadh)

إن الجدول رقم (٦-٣) ليس بالشكل الطبيعى الأول؛ لأن حقل الموقع فى السجل الثانى يحتوى على أكثر من قيمة. ويمكن تفسير محتويات حقل «الموقع» وفق أحد التفسيرين التاليين:

١- مدى حقل «الموقع» مكون من قيم غير مركبة (وهى أسماء المدن)، ولكن الحقل قد يحتوى على أكثر من قيمة. ويعنى هذا أن حقل «الموقع» لا يعتمد وظيفياً على حقل «رقم القسم». والسبب وراء ذلك أن قيمة المفتاح الرئيسى لا يمكن أن تحدد قيمة واحدة لحقل «الموقع».

٢- مدى حقل «الموقع» يتكون من مجموعة من القيم، وبذلك فهو ذو قيم مركبة. وفى هذه الحالة فإن حقل «الموقع» يعتمد وظيفياً على المفتاح الرئيسى للجدول.

ووفقاً لكلا التفسيرين السابقين لحقل «الموقع» لا يعد الجدول رقم (٦-٣) بالشكل الطبيعى الأول (أو جدولاً علاقياً). ولتطبيع الجدول حتى يصبح بالشكل الطبيعى الأول، يوجد ثلاثة طرق، وهى كما يلي:

١- إزالة حقل «الموقع» الذى يخالف الشكل الطبيعى الأول ووضعه فى جدول جديد. ويضاف للجدول الجديد حقل المفتاح الرئيسى للجدول الأصلى بحيث يصبح المفتاح الرئيسى للجدول الجديد مكوناً من حقلين هما حقل «رقم القسم» وحقل «الموقع»، وبحيث يوجد سجل لكل موقع من مواقع أى قسم فى الجدول الجديد. ويكون الجدولان الناتجان كما هو موضح فى الشكل رقم (٦-٣).

شكل رقم (٦-٣): نتيجة التطبيع للشكل الطبيعى الأول وفق الطريقة الأولى

DEP_LOCATION

DNO	Location
10	Riyadh
20	Riyadh
20	Jeddah
20	Dammam
30	Riyadh

DEPARTMENT

DNO	DName	D_MGR_NO
10	Research	10101010
20	Computer Center	20202020
30	Administration	30303030

٢- إضافة حقل «الموقع» ضمن المفتاح الرئيسى للجدول الأصلى بحيث يوجد سجل لكل موقع من مواقع القسم كما هو موضح فى الشكل (٦-٤).

شكل رقم (٦-٤): نتيجة التطبيع للشكل الطبيعى الأول وفق الطريقة الثانية

DEPARTMENT

DNO	DName	D_MGR_NO	Location
10	Research	10101010	Riyadh
20	Computer Center	20202020	Riyadh
20	Computer Center	20202020	Jeddah
20	Computer Center	20202020	Dammam
30	Administration	30303030	Riyadh

٣- إضافة حقول جديدة للجدول الأصلى تساوى الحد الأعلى لعدد مواقع الأقسام المسموح به. فلو افترضنا أن الحد الأعلى لعدد مواقع أى قسم هو ثلاثة، يمكن إعادة تصميم الجدول ليصبح متوافقاً مع الشكل الطبيعى الأول وفق هذه الطريقة كما هو موضح فى الشكل رقم (٦-٥).

شكل رقم (٦-٥): نتيجة التطبيع للشكل الطبيعى الأول وفق الطريقة الثالثة

DEPARTMENT

DNO	DName	D_MGR_NO	Location1	Location2	Location3
10	Research	10101010	Riyadh		
20	Computer Center	20202020	Riyadh	Jeddah	Dammam
30	Administration	30303030	Riyadh		

إن الخيار الأفضل من ضمن الخيارات الثلاثة أعلاه هو الخيار الأول: لأنه لا يؤدى إلى تكرارية فى البيانات كما هو الحال فى الخيار الثانى، كما أنه لا يضيع المساحة التخزينية أو يقيد الحد الأعلى من المواقع كما هو الحال فى الخيار الثالث. بالإضافة إلى ذلك، لو تم اختيار الطريقة الثانية، فإن الجدول سيتم تقسيمه إلى جدولين ليصبح كما فى الطريقة الأولى فى أثناء عمليات تطبيع الجدول فى مراحل لاحقة. أما الطريقة الثالثة فمن عيوبها أيضاً، مقارنة بالخيار الأول، هو أنها تعقد إجراء عمليات التعامل مع الجدول. فعلى سبيل المثال، كيف ستتم كتابة تعليمة الاستفسار المكافئة للاستفسار التالى: «ما الأقسام التى يوجد لها مواقع فى مدينة جدة؟».

ويمنع الشكل الطبيعى الأول من أن تكون قيم الحقول متعددة القيم ومركبة فى آن واحد. فعلى سبيل المثال، يوضح الجدول رقم (٦-٤) بيانات الموظفين وبيانات المشاريع

التى يعملون عليها. فكل موظف يعمل على عدد من المشاريع، وكل مشروع يعمل عليه موظف له رقم وعدد من الأسابيع التى عملها الموظف على المشروع.

جدول رقم (٦-٤): جدول يحتوى على حقول متعددة القيم ومركبة

EMP_PROJ

ENO	EName	Title	Salary	Project	
				PNO	Weeks
E1	Saleh Aloufi	Electrical Eng	40000	P1	12
E2	Ahmad Alhamid	System Analyst	35000	P3	20
				P2	10
E3	Mohamed Alhamad	Mechanical Eng.	37000	P3	10
				P4	14
E4	Khalid Alsaleh	Programmer	29000	P2	26
E5	Ibraheem Alotaibi	Database Admin.	45000	P2	14
E6	Mishal Aleesa	Programmer	29000	P4	16
E7	Abdullah Alghanim	Programmer	29000	P3	12
E8	Turky Alsalman	Secretary	25000	P3	12

فى الجدول السابق يعمل كل من الموظف رقم "E2" والموظف رقم "E3" على مشروعين. وبيانات كل مشروع متعددة القيم لكون كل من هذين الموظفين يعمل على أكثر من مشروع، وفى الوقت نفسه، مركبة من حقل «رقم المشروع» وحقل «عدد الأسابيع». ويمثل «رقم الموظف» المفتاح الرئيسى للجدول، لكونه يميز بين سجلات الموظفين المختلفة، فى حين يمثل «رقم المشروع» مفتاحاً جزئياً يميز بين المشاريع المختلفة التى يعمل عليها الموظف نفسه. ولأن الجدول السابق ليس فى الشكل الطبيعى الأول، فإنه يمكن تطبيعه ليصبح فى الشكل الطبيعى الأول من خلال تجزئته إلى جدولين: جدول خاص ببيانات الموظفين، وجدول خاص ببيانات المشاريع التى تمثل الحقل المتعدد القيم والمركب. ويصبح المفتاح الرئيسى للجدول الجديد، الذى يمثل بيانات المشاريع التى يعمل عليها الموظفون، مكوناً من المفتاح الرئيسى للجدول الأصلى بالإضافة إلى حقل «رقم المشروع». وبذلك يمكن الربط بين الجدولين ومعرفة بيانات المشاريع التى يعمل عليها كل موظف. ويمثل الشكل رقم (٦-٦) الجدولين الناتجين بعد إجراء عملية التطبيع على الجدول الأصلى.

شكل رقم (٦-٦): تطبيع جدول ذي حقل متعدد القيم ومركب للشكل الطبيعى الأول

EMPLOYEE

ENO	EName	Title	Salary
E1	Saleh Aloufi	Electrical Eng	40000
E2	Ahmad Alhamid	System Analyst	35000
E3	Mohamed Alhamad	Mechanical Eng.	37000
E4	Khalid Alsaleh	Programmer	29000
E5	Ibraheem Alotaibi	Database Admin.	45000
E6	Mishal Aleesa	Programmer	29000
E7	Abdullah Alghanim	Programmer	29000
E8	Turky Alsalman	Secretary	25000

EMP_PROJECTS

ENO	PNO	Weeks
E1	P1	12
E2	P3	20
E2	P2	10
E3	P3	10
E3	P4	14
E4	P2	26
E5	P2	14
E6	P4	16
E7	P3	12
E8	P3	12

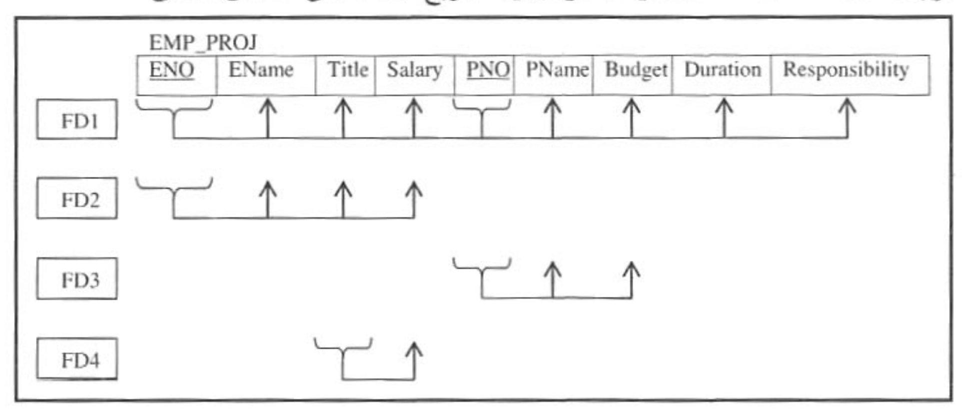
٦-١-٣-٢ الشكل الطبيعى الثانى ((Second Normal Form (2NF)):

يعتمد الشكل الطبيعى الثانى على مبدأ الاعتمادية الوظيفية الكاملة (Full Functional Dependency). وتسمى أية اعتمادية وظيفية $(X \rightarrow Y)$ اعتمادية وظيفية كاملة إذا كان من غير الممكن إزالة أى حقل من الحقول المكونة للجانب الأيسر من الاعتمادية مع استمرار تحديد الاعتمادية للجانب الأيمن. ويعنى هذا أن عدد حقول الجانب الأيسر يعد أقل عدد ممكن من الحقول التى تمكن من تحديد الجانب الأيمن فى الاعتمادية. أما إذا كان الأمر غير ذلك، فإن الاعتمادية الوظيفية تعد جزئية (Partial Functional Dependency) بمعنى أنه يمكن الاستغناء عن حقل أو أكثر من حقول الجانب الأيسر مع الاستمرار فى تحديد الجانب الأيمن فى الاعتمادية الوظيفية. ويمكن تعريف هذين النوعين من الاعتماديات الوظيفية بشكل رسمى كما يلى:

- إذا وجدت اعتمادية وظيفية $(X \rightarrow Y)$ فإنها تعد اعتمادية وظيفية كاملة $(X \xrightarrow{f} Y)$ إذا كان من غير الممكن إزالة أى حقل من الحقول المكونة للجانب الأيسر من الاعتمادية، وهو A وبحيث إن $(A \in X)$، مع الاستمرار فى تحديد الجانب الأيمن من الاعتمادية $(X\text{-}\{A\} \nrightarrow Y)$.

- إذا وجدت اعتمادية وظيفية $(X \rightarrow Y)$ فإنها تعد اعتمادية وظيفية جزئية $(X \xrightarrow{p} Y)$ إذا كان من الممكن إزالة أى حقل من الحقول المكونة للجانب الأيسر من الاعتمادية، وهو (A) وبحيث إن $(A \in X)$، مع الاستمرار فى تحديد الجانب الأيمن من الاعتمادية $(X\text{-}\{A\} \rightarrow Y)$.

ففى جدول «الموظفين - المشـــاريع» (EMP_PROJ) الممثــل فى الجدول رقم (٦-٢) توجد، كما أسلفنا، الاعتماديات الوظيفية الأربع الممثلة فى الشكل التالى:



إن الاعتمادية الوظيفية الأولى أعلاه (FD1) تدل على أن المفتاح الرئيسى المكون من حقل «رقم الموظف» وحقل «رقم المشـــروع» يحددان قيم الحقول كافة فى أى سجل من ســـجلات الجدول. وهذه الخاصية هى الخاصية الرئيسية للمفتاح الرئيسى حيث إن قيمته تحدد السجل المطلوب بشكل منفرد وقيم حقوله كافة. أما الاعتمادية الوظيفية الثانية والاعتمادية الوظيفية الثالثة فهى اعتماديات وظيفية جزئية، وذلك لأن كلاً منها يمكن اشـــتقاقها من الاعتمادية الوظيفية الأولى. ففـــى الاعتمادية الوظيفية الثانية، تعتمد قيم حقل اسم الموظف، وحقل مسماه الوظيفى، وحقل راتبه على رقم الموظف دون الحاجة إلى معرفة رقم المشروع. ويعنى هذا أنه يمكن معرفة قيم هذه الحقول الثلاثة دون معرفة رقم المشروع الذى يعمل عليه الموظف. أما فى الاعتمادية الوظيفية الثالثة، فتعتمد قيمة حقل اســـم المشروع، وحقل ميزانيته على قيمة رقم المشروع دون الحاجة إلـــى معرفة رقم الموظف الذى يعمل فيه. لذا فإن كلاً من الاعتمادية الوظيفية الثانية والإعتمادية الوظيفية الثالثة تعدان اعتماديات وظيفية جزئية من الاعتمادية الوظيفية الأولى. وحسب تعريف الاعتماديات الوظيفية أعلاه، فإن الاعتمادية الوظيفية الأولى تعـــد اعتمادية وظيفية جزئية لوجود اعتماديات وظيفية أخرى يمكن أن تشـــتق منها. أما الاعتمادية الوظيفية الرابعة فهى اعتمادية وظيفية كاملة؛ إذ لا يمكن أن تشـــتق أو تســـتنتج من الوظائف الاعتمادية الأخرى. وبالنظـــر فى البيانات المدونة فى الجدول نلاحظ أن هذه الاعتماديات، الثانية والثالثة والرابعة، دائماً متحققة.

وقد يطرح السـؤال التالى: كيف نسـتطيع أن نتعرف على الاعتماديات الوظيفية؟ والإجابة هى أن الاعتماديات الوظيفية لا تستنتج من قبل مصممى قواعد البيانات من خلال النظر إلى البيانات التى سـيتم تخزينها فى جـداول قاعدة البيانات، وإنما يتم التعرف عليها من خلال المسـتفيدين من قاعدة البيانات. فالمسـتفيدون العاملون فى الشـئون الإدارية فى المنظمة، على سـبيل المثال، قد يوضحون أنه من الممكن التعرف علـى بيانات أى موظف من خـلال معرفة رقمه الوظيفى. ويعنى هذا وجود اعتمادية وظيفيـة بين رقم الموظف وبقيـة البيانات الوظيفية الخاصة فيـه. كذلك هو الحال بالنسـبة للمرضى المنومين فى مستشفى ما، فقد يفيد العاملين فى المستشفى أن رقم تحويلة هاتف المريض تدل على رقم الغرفة أو السـرير المنوم فيه المريض. ويعنى هذا أن رقم غرفة أو سرير المريض يعتمد وظيفياً على رقم تحويلة هاتفه.

وللحد من تكرارية البيانات التى تؤدى إلى مشكلات التعديل، يعتمد تعريف الشكل الطبيعى الثانى على عدم وجود أى اعتماديات وظيفية جزئية كما يلى:

> يعد هيكل أى جدول علاقى فى شكله الطبيعى الثانى (2NF) إذا كان فى الشكل الطبيعى الأول، وكان كل حقل من حقوله غير الأولية يعتمد كلياً على المفتاح الرئيسى للجدول.

يعتمـد التعريـف أعلاه على وجود مفتاح مرشـح واحد هو المفتاح الرئيسـى وأن حقوله الأولية هى مجموعة حقول المفتاح الرئيسى فحسب. أما التعريف الأعم للشكل الطبيعى الثانى فيأخذ بعين الاعتبار وجود مفاتيح مرشحة أخرى، ومن ثم وجود حقول أوليـة غيـر تلك الحقول التى يتكون منها المفتاح الرئيسـى. وهذا التعريف العام كما يلى:

> يعد هيكل أى جدول علاقى فى شكله الطبيعى الثانى (2NF) إذا كان فى الشكل الطبيعى الأول، وكان كل حقل من حقوله غير الأولية (بمعنى أن الحقل ليس جزءاً من أى مفتاح مرشح) يعتمد كلياً على كل مفتاح مرشح للجدول.

ويعنى التعريف الأعم للشـكل الطبيعى الثانى أن جميع الحقول غير الأولية يجب أن تعتمـد وظيفياً على جميع المفاتيح المرشـحة وليس على المفتاح الرئيسـى للجدول

فقط. وللتحقق من كون أى جدول فى شكله الطبيعى الثانى، حسب التعريف الأول، يتم اختبار الاعتماديات الوظيفية المفروضة عليه والتى يكون جانبها الأيسر جزءاً من المفتاح الرئيسى للجدول. ويكون الجدول فى شكله الطبيعى الثانى إذا كان فى شكله الطبيعى الأول وتحققت فيه أى من الشروط الثلاثة التالية:

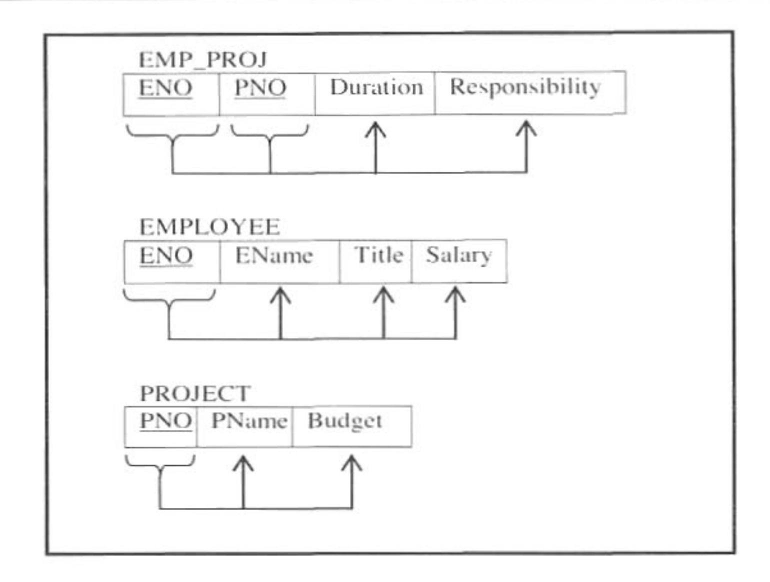
١- المفتاح الرئيسى مكون من حقل واحد فقط.

٢- جميع حقول الجدول تعد جزءاً من مفتاحه الرئيسى بمعنى عدم وجود أى حقل غير أوَّلى.

٣- كل حقل ليس من حقول المفتاح الرئيسى يعتمد على جميع حقول المفتاح الرئيسى وليس على جزء منها.

أما إذا احتوى الجدول على مفاتيح مرشحة أخرى غير المفتاح الرئيسى له، فإنه يجب التحقق من أن كل حقل غير أولى يعتمد كلياً على كل مفتاح مرشح، بالإضافة إلى اعتماده الكلى على المفتاح الرئيسى.

ويعد الجدول «الموظف - المشروع» (EMP_PROJ) الممثل فى جدول رقم (٦-٢) ليس فى الشكل الطبيعى الثانى، وذلك بسبب الاعتمادية الوظيفية الثانية والاعتمادية الوظيفية الثالثة اللتين تعتمد أجزاؤهما اليمنى على جزء من حقول المفتاح الرئيسى، وليس حقوله كافة. ولهذا السبب تتكرر بعض بيانات الجدول مما يؤدى إلى أخطاء التعديل على الجدول التى سبق أن أوضحناها أعلاه. ولتطبيع الجدول بحيث يصبح فى شكله الطبيعى الثانى، تتم تجزئة الجدول إلى اثنين أو أكثر من الجداول بحيث ينطبق على كل منها أحد الشروط الثلاثة أعلاه. وبمعنى آخر، تتم تجزئة الجدول إلى مجموعة من الجداول تنطبق عليها شروط الشكل الطبيعى الثانى. ويتم ذلك من خلال إنشاء جدول جديد لكل اعتمادية وظيفية جزئية حيث يتم إنشاء جدول اسمه «موظف» (EMPLOYEE)، فى مثالنا، لتمثيل جميع حقول الاعتمادية الوظيفية الثانية، وجدول «مشروع» (PROJECT) لتمثيل جميع حقول الاعتمادية الوظيفية الثالثة، مع الإبقاء على بقية الحقول فى الجدول الأساسى دون تغيير لها. وتكون أشكال الجداول الناتجة بعد عملية التجزئة جداول بالشكل الطبيعى الثانى تعتمد حقولها غير الأولية على جميع حقول مفاتيحها الرئيسية، كما يلى:



٦-١-٣-٣ الشكل الطبيعى الثالث ((Third Normal Form (3NF)):

يعتمد تطبيع الجداول إلى الشكل الطبيعى الثالث على مبدأ الاعتمادية الوظيفية الانتقالية (Transitive Dependency). وتعد الاعتمادية الوظيفية $(X \rightarrow Y)$ فى جدول ما اعتمادية وظيفية انتقالية إذا وجد مجموعة من الحقول، ولتكن (Z) فى الجدول، ولا تمثل مفتاحاً مرشحاً للجدول كما أنها ليست مجموعة جزئية من أى مفتاح (سواء كان مرشحاً أو رئيسياً) للجدول مع وجود الاعتمادية الوظيفية $(X \rightarrow Z)$ والاعتمادية الوظيفية $(Z \rightarrow Y)$. ويمكن تعريف الاعتمادية الوظيفية الانتقالية بشكل رسمى، كما يلى:

> توجد اعتمادية وظيفية انتقالية $(X \rightarrow Y)$ إذا تحققت الشروط التالية:
>
> ١- $X \rightarrow Z$
>
> ٢- $Z \rightarrow Y$
>
> ٣- $Z \nrightarrow X$
>
> ٤- $Y \not\subseteq Z$

وتعنى الاعتمادية الانتقالية $(X \rightarrow Y)$، بشكل عام، أنه يمكن تحديد حقول الجانب الأيمن من الاعتمادية من خلال اعتماديات وظيفية أخرى عوضاً عن هذه الاعتمادية التى تحدد حقول الجانب الأيمن بشكل مباشر.

ولتعريف الشـــكل الطبيعى الثالث والشكل الطبيعى «بويس-كود» (Boyce-Codd Normal Form (BCNF)، نحتاج إلى تعريف الاعتمادية الوظيفية البديهية، وهو كما يلى:

> تعد الاعتمادية الوظيفية $X \rightarrow Y$ بديهية إذا كانت الحقول المكونة للجانب الأيمن من الاعتمادية الوظيفية مجموعة جزئية أو مساوية لحقول الجانب الأيسر من الاعتمادية ($Y \subseteq X$).

ومثـــال على الاعتماديات الوظيفية البديهية من جدول «الموظف - المشـــروع»، تعد الاعتماديات الوظيفية التالية بديهية:

١- ENO, PNO → ENO, PNO

٢- ENO, PNO → ENO

٣- ENO, PNO → PNO

٤- PNO → PNO

٥- ENO → ENO

وحســـب التعريف أعـــلاه، تعد كل اعتماديـــة وظيفية من الاعتماديـــات الوظيفية الخمس الســـابقة بديهية؛ لأن حقول الجانب الأيمـــن هو مجموعة جزئية أو مجموعة مساوية لحقول الجانب الأيسر. فعلى سبيل المثال، تنص الاعتمادية الوظيفية الرابعة على أن رقم المشـــروع يحدد رقم المشـــروع، وهو أمر بديهى. كذلك هو الحال بالنسبة للاعتمادية الوظيفية الثانية، على ســـبيل المثال، التى تنص على أن رقم الموظف ورقم المشـــروع يحددان رقم الموظف، وهو أمـــر بديهى حيث يمكن تحديد رقم الموظف حتى بدون معرفة رقم المشروع.

وفيما يلى تعريف الشـــكل الطبيعى الثالث بشـــكله الأصلى حسب ما اقترحه «كود» (Codd, 1972):

> يعد هيكل أى جدول علاقى فى شكله الطبيعى الثالث (3NF) إذا كان فى الشكل الطبيعى الثانى، وكان كل حقل من حقوله غير الأولية لا يعتمد بشكل انتقالى على المفتاح الرئيسى للجدول.

وكما هو الحال بالنسبة للتعريف العام للشكل الطبيعى الثانى، يفترض التعريف العام للشـــكل الطبيعى الثالث وجود أكثر من مفتاح مرشـــح فى الجدول، عوضاً عن افتراض وجود مفتاح مرشح واحد وهو المفتاح الرئيسى للجدول. وهذا التعريف العام كما يلى:

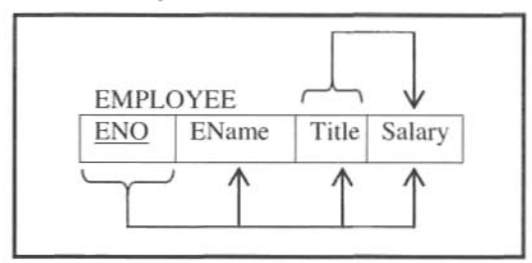
إذا افترضنا وجود جدول اسمه "R" يحتوى على عدد من الحقول "A" بحيث إن $A = (A_1, A_2, ..., A_n)$، يكون الجدول فى شكله الطبيعى الثالث (3NF) إذا كانت جميع الاعتماديات الوظيفية المفروضة عليه بالشكل $(X \rightarrow Y)$، وبحيث إن كلاً من "X" و "Y" تمثل مجموعة جزئية من حقول الجدول $(X \subseteq A, Y \subseteq A)$، فإنه يجب أن تتحقق على كل اعتمادية وظيفية، على الأقل أحد الشروط التالية:

١- $X \rightarrow Y$ اعتمادية وظيفية بديهية.

٢- X عبارة عن مفتاح خارق للجدول.

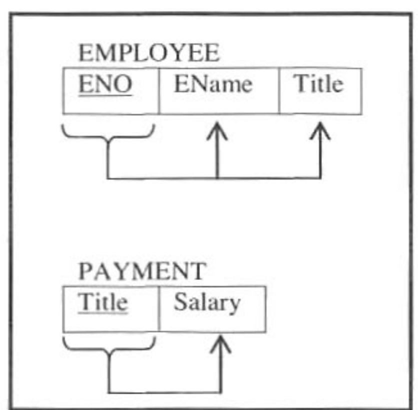
٣- Y عبارة عن حقل أولى (أو مجموعة حقول أولية).

ويلاحظ فى التعريف العام للشكل الطبيعى الثالث على أنه لا ينص على أن يكون الجدول فى شكله الطبيعى الثانى، وذلك لكون هذا التعريف يمنع وجود اعتماديات وظيفية جزئية، التى ينص على عدم وجودها تعريف الشكل الطبيعى الثانى، بشكل مباشر. وعند تطبيق التعريف الأول للشكل الطبيعى الثالث، الذى ينص على أن يكون الجدول فى الشكل الطبيعى الثانى، على الجداول الثلاثة التى نتجت بعد عملية تطبيع جدول «مشروع - موظف» إلى الشكل الطبيعى الثانى، نجد أنه لا يوجد اعتماديات وظيفية انتقالية فى كل من الجدول الأول (EMP_PROJ) والجدول الثالث (PROJECT). لذا فإن هذين الجدولين هما فى الشكل الطبيعى الثالث أيضاً. أما الجدول الثانى (EMPLOYEE) فليس فى الشكل الطبيعى الثالث، وذلك لوجود الاعتمادية الوظيفية الرابعة التى تمثل اعتمادية وظيفية انتقالية؛ إذ إنها تمكن من تحديد قيمة حقل «الراتب» ليس من خلال معرفة قيمة حقل المفتاح الرئيسى للجدول، وهو «رقم الموظف»، بشكل مباشر فحسب، ولكن يمكن أيضاً تحديده، بشكل انتقالى، من خلال معرفة «مسمى الوظيفة (رقم الوظيفة ← مسمى الوظيفة ← الراتب)، كما يلى:



ومثل هذه الاعتمادية الانتقالية تؤدى إلى مشكلات التعديل التى سبق أن أوضحناها بسبب تكرارية البيانات، إذ إننا سنجد أنه كلما تكرر مسمى وظيفة معينة تكرر راتب هذه الوظيفية. وللتغلب على هذه التكرارية، يتم تقسيم الجدول إلى أكثر

من جدول حسب عدد الاعتماديات الوظيفية الانتقالية بحيث يتم إنشاء جدول جديد لكل اعتمادية وظيفية انتقالية. ولأنه يوجد فى مثالنا اعتمادية وظيفية انتقالية واحدة فقط فإنه يتم إنشاء جدول جديد اسمه «الرواتب» (PAYMENT) تكون حقوله مكونة من الحقول الواردة فى الاعتمادية وهى حقل «مسمى الوظيفة» وحقل «الراتب» بحيث يكون المفتاح الرئيسى للجدول الجديد هو الحقل الموجود (أو مجموعة الحقول الموجودة) فى الجانب الأيسر من الاعتمادية، كما يلى:



أما إذا طبقنا التعريف العام للشكل الطبيعى الثالث مباشرة على أى جدول دون تطبيع الجدول إلى الشكل الطبيعى الثانى، فإن هذا التعريف سيمكننا من معرفة الاعتماديات الوظيفية التى تخالف شروط الشكل الطبيعى الثانى، بالإضافة لشروط الشكل الطبيعى الثالث. فعلى سبيل المثال، لو تم تطبيق التعريف العام على جدول «الموظف - المشروع» سنجد أن الاعتمادية الوظيفية الثانية (ENO → (EName, Title, Salary)) والاعتمادية الوظيفية الثالثة (PNO → (PName, Budget)) تخالفان شروط الشكل الطبيعى الثانى، وذلك لكونهما جزئية، فهما تخالفان أيضاً شروط الشكل الطبيعى الثالث وذلك لكونهما ليستا بديهيتين، وليست الأجزاء اليسرى منهما تمثل مفاتيح خارقة للجدول، وليست الأجزاء اليمنى منهما تمثل حقولاً أولية. لذا سيتم بناء جداول جديدة لهاتين الاعتماديتين كما هو الحال بالنسبة للاعتمادية الرابعة. وتكون نتيجة عملية التطبيع هذه مماثلة لعملية تطبيع الجدول إلى الشكل الطبيعى الثانى، ومن ثم تطبيعه إلى الشكل الطبيعى الثالث. ويدل هذا على أنه ليس من الضرورى أن تمر عملية التطبيع بالشكل الطبيعى الثانى، ولكنه يمكن تطبيع الجدول مباشرة (من الشكل الطبيعى الأول) إلى الشكل الطبيعى الثالث، إلا أن عملية التسلسل فى مراحل التطبيع ذات بعد تاريخى فقط لكون «كود» قد طرحها بهذه الطريقة.

٦-١-٣-٤ الشكل الطبيعى «بويس - كود» (Boyce-Codd Normal Form (BCNF)):

تم اقتراح الشكل الطبيعى «بويس - كود» بشكل أساسى على أنه شكل مبسط للشكل الطبيعى الثالث. إلا أنه تبين لاحقاً أن هذا الشكل أشد فى شروطه من الشكل الطبيعى الثالث، بمعنى أنه إذا كان أى جدول فى الشكل الطبيعى «بويس - كود»، فإنه أيضاً فى الشكل الطبيعى الثالث والعكس ليس بالضرورة صحيحاً: إذ إن جدولاً ما قد يكون فى الشكل الطبيعى الثالث، ولكنه ليس فى الشكل الطبيعى «بويس - كود». إن تعريف الشكل الطبيعى الثالث لا يلغى جميع الاعتماديات الوظيفية الانتقالية وبوجه خاص تلك التى يكون جانبها الأيمن حقلاً أولياً (أو مجموعة حقول أولية). فعلى سبيل المثال، لنفترض أن جدول «الموظف - المشروع» كان يحتوى على حقل يسمى «موقع المشروع» (PLocation)، وأن كل مشروع قد يكون له عدد من المواقع وليس موقعاً واحداً فحسب. ولنفترض أيضاً وجود الاعتماديتين الوظيفيتين التاليتين:

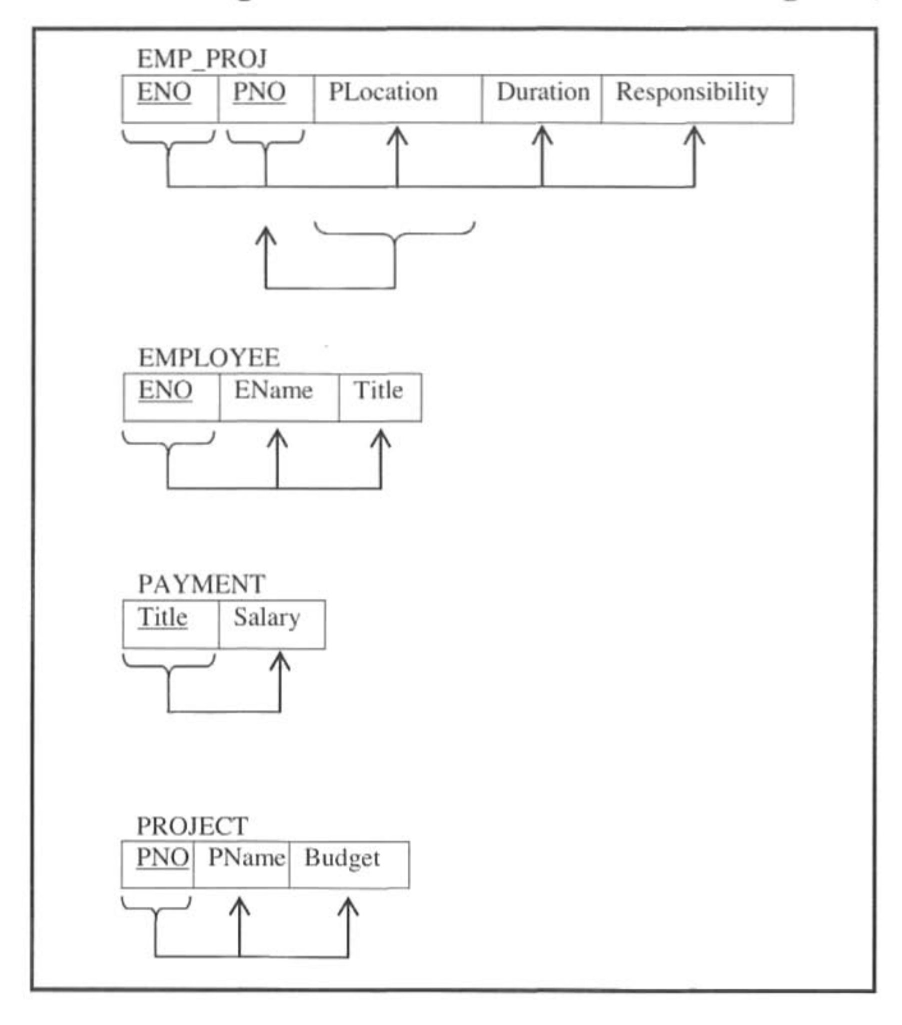
- قد يعمل الموظف فى أكثر من مشروع، ولكنه عندما يعمل فى مشروع، يكون عمله فى موقع واحد من مواقع المشروع وأن للموظف مسئولية محددة فى المشروع وفترة زمنية تحدد فترة عمله فى المشروع (ENO, PNO → PLocation, Duration, Responsibility).
- يوجد فى أى موقع مشروع واحد فقط (PLocation → PNO).

وبعد إضافة الحقل الجديد للجدول وفق الاعتماديتين الوظيفيتين أعلاه، يصبح شكل الجدول كما يلى:

EMP_PROJ

ENO	EName	Title	Salary	PNO	PName	PLocation	Budget	Duration	Responsibility
E1	Saleh Aloufi	Electrical Eng	40000	P1	Database System	Riyadh	170000	12	Manager
E2	Ahmad Alhamid	System Analyst	35000	P3	Human Resources	Jizan	190000	20	System Analyst
E2	Ahmad Alhamid	System Analyst	35000	P2	Inventory	Jouf	220000	10	System Analyst
E3	Mohamed Alhamad	Mechanical Eng.	37000	P3	Human Resources	Joubail	190000	10	Consultant
E3	Mohamed Alhamad	Mechanical Eng.	37000	P4	Maintenance	Skaka	230000	14	Consultant
E4	Khalid Alsaleh	Programmer	29000	P2	Inventory	Jouf	220000	26	Programmer
E5	Ibraheem Alotaibi	Database Admin.	45000	P2	Inventory	Jouf	220000	14	Manager
E6	Mishal Aleesa	Programmer	29000	P4	Maintenance	Hail	230000	16	Programmer
E7	Abdullah Alghanim	Programmer	29000	P3	Human Resources	Jizan	190000	12	Manager
E8	Turky Alsalman	Secretary	25000	P3	Human Resources	Jizan	190000	12	Project Secretary

ونتيجـــة للاعتماديـــة الوظيفية الثانية أعلاه، وكما يلاحظ فـــى البيانات المدونة فى الجدول، يمكن تحديد رقم أى مشـــروع من خلال معرفة «الموقع»؛ لأنه لا يمكن أن يوجد مشروعان فى موقع واحد. وبناء على تعريف الشكل الطبيعى الثالث، تتم تجزئة الجدول إلى أربعة جداول هى: جدول «الموظف»، وجدول «المشروع»، وجدول «الرواتب»، وجدول «الموظف - المشـــروع» الذى يحتوى على بقية الحقول التى لم تنقل إلى أى من الجداول الثلاثة الأخرى بالإضافة للمفتاح الرئيسى للجدول المكون من حقل «رقم الموظف» وحقل «رقم المشروع»، كما سبق أن أوضحنا فى الجزء السابق، لتصبح كما يلى:



ويلاحــظ أن الاعتماديــة الوظيفية التى تحدد رقم المشــروع من خلال الموقع، فى الجدول الأول أعلاه، تعد انتقالية ولكنها لا تخالف شــروط الشــكل الطبيعى الثالث لكــون الجانب الأيمن منهــا يمثل حقلاً أولياً (جزءاً من المفتاح الرئيســى للجدول فى هذه الحالة). لذا فإن الجدول «الموظف - المشــروع» يعد فى الشكل الطبيعى الثالث. إلا أن وجود هذه الاعتمادية فيه يؤدى إلى تكرارية فى بياناته. فعلى ســبيل المثال، لو افترضنا وجود عشــرين موقعاً لخمســة من المشاريع التى تقوم المنظمة بالتعامل معها (أو متابعتها) ووجود عشرة آلاف موظف يعملون فى هذه المشاريع الخمسة، فإن هذين الحقلين ســتكرر قيمهما بشــكل كبير ضمن الجدول. وللحد من هــذه التكرارية فى بيانات الجدول، يمكن إنشــاء جدول جديد يحتوى على قيم أرقام المشــاريع الخمسة، والعشــرين موقعــاً التى توجد فيها هذه المشــاريع. لذا فإن التعريف التالى للشــكل الطبيعى «بويس - كود» يحد من مثل تكرارية البيانات هذه ويعد أشد فى شروطه من شروط الشكل الطبيعى الثالث:

> إذا افترضنا وجود جدول اسمه "R" يحتوى على عدد من الحقول "A" بحيث إن $A = (A_1, A_2, ..., A_n)$، يكون الجدول فى شكله الطبيعى بويس - كود (BCNF) إذا كانت جميع الاعتماديات الوظيفية المفروضة عليه بالشكل $(X \rightarrow Y)$، وبحيث إن كل من "X" و "Y" تمثل مجموعة جزئية من حقول الجدول $(X \subseteq A, Y \subseteq A)$، فإنه يجب أن تتحقق على كل اعتمادية وظيفية، على الأقل أحد الشروط التالية:
>
> ١- $X \rightarrow Y$ اعتمادية وظيفية بديهية.
>
> ٢- X عبارة عن مفتاح خارق للجدول.

تنطبق شروط الشكل الطبيعى «بويس - كود» على الجدول الثانى والثالث والرابع، لأن أى اعتمادية وظيفية (غير بديهية) فى هذه الجداول يكون طرفها الأيســر مفتاحاً خارقاً (وهو المفتاح الرئيسى للجدول). إلا أن الجدول الأول ليس فى الشكل الطبيعى «بويس - كود» لوجود الاعتمادية الوظيفية الجديدة التى تحدد قيمة حقل «رقم المشروع» من قيمة حقل «الموقع» وطرفها الأيســر، «الموقع» ليس مفتاحــاً خارقاً للجدول. ولتطبيع الجدول بحيث يصبح فى شــكله الطبيعى «بويس - كود»، يمكن تجزئة الجدول إلى جدولين حتى يتوافق مع شــروط الشــكل الطبيعى «بويس - كود». إلا أن عملية تجزئة الجدول ليست بديهية؛ إذ هناك ثلاثة بدائل لعملية التجزئة، وهذه البدائل الثلاثة كما يلى:

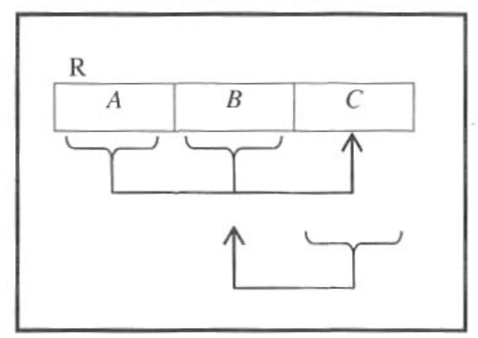
١- (ENO, PNO, Duration, Responsibility) و (ENO, PLocation)

٢- (PNO, ENO, Duration, Responsibility) و (PNO, PLocation)

٣- (PLocation, ENO, Duration, Responsibility) و (PLocation, PNO)

ويوضح الشـــكل رقم (٦-٧) الشـــكل العام للاعتمادية الوظيفية التى تخل بشروط الشكل الطبيعى «بويس - كود».

شكل رقم (٦-٧): الشكل العام للاعتمادية الوظيفية التى تخل بشروط الشكل الطبيعى بويس - كود



وتعنى الاعتمادية الوظيفية أعلاه وجود مفتاح مرشح (أو رئيسي) يتمثل فى الحقلين (A,B) يحددان قيمة الحقل (C). كما يوجد فى الوقت نفسه اعتمادية وظيفية هى أن الحقــل (C) يحــدد قيمة حقل أولى وهو الحقــل (B) (أو (A)). ونظراً لأن الحقل فى الجانــب الأيمن من الاعتمادية هو حقل أولى فإن هذه الاعتمادية لا تخالف شــروط الشــكل الطبيعى الثالث بينما تخالف شــروط الشــكل الطبيعى «بويس - كود» لكون الحقل فى الجانب الأيسر من الاعتمادية وهو الحقل (C) ليس مفتاحاً خارقاً للجدول. ولجعل الجدول فى الشكل الطبيعى «بويس - كود»، يتم تجزئته إلى جدولين، حسب ما أسلفنا أعلاه، وفق أحد البدائل (أو التوليفات) الثلاثة الممكنة، وهى:

١- (A, B) و (A,C)

٢- (B,A) و (B,C)

٣- (C, A) و (C,B)

ويلاحــظ أن كلا الحقلين فى الجداول الموجودة فى الجهــة اليمنى من التوليفات الثلاثة أعلاه يمثلان المفتاح الرئيسى للجدول الأول (وذلك إذا كان الحقل (A) والحقل (B) مفتاحاً رئيســياً للجدول الأصلى وليس مفتاحاً مرشــحاً). كما يلاحظ أن المفتاح الرئيســى للجدول الثانى فى البديل الأول يتكون مــن الحقلين (A) و (C)، وذلك لعدم وجــود اعتمادية وظيفية بين الاثنين. أما المفتاح الرئيســى للجدول الثانى فى كل من

البديـل الثانى والبديل الثالث فيتكون من حقـل واحد هو الحقل (C)؛ وذلك لأن قيمة هـذا الحقل تحدد قيمة الحقل الآخـر وفقاً للاعتمادية الوظيفية المخالفة لشـروط الشـكل الطبيعى «بويس - كود». وفى كل البدائـل الثلاثة أعلاه، تدرج بقية الحقول الموجودة فى الجدول الأصلى (إن وجدت بالإضافة للحقول الثلاثة المدونة فى الجدول أعلاه) ضمن حقول الجداول الموجودة فى الجهة اليمنى من البدائل، وذلك لكون بقية الحقول هذه تحتاج إلى مفتاح رئيسى يتكون من حقلين لتحديد قيمهما، كما فى المثال السابق.

وفـى كل من التوليفات الثلاثة أعلاه يتم فقد الاعتمادية الوظيفية التى تنص على أن الحقـل (A) والحقـل (B)، مجتمعين، يحددان قيمة الحقل (C): A, B → C . أما فى مثالنـا أعلاه، فإنه يتم فقد الاعتمادية الوظيفيـة (ENO, PNO → PLocation). وتعنى كلمـة «فقد» أنه لا يمكن فرض هـذه الاعتمادية بعد تجزئة الجـدول كما هو الحال قبل تجزئته. وتسـمى خاصية المحافظـة على الاعتماديات الوظيفيـة بعد التجزئة بخاصية «المحافظة علـى الاعتماديات» (Dependency Preservation). ومن بين البدائل الثلاثة أعلاه، وعلى الرغم من أنها جميعاً ستفقدنا القدرة على فرض قيد الإعتمادية الوظيفية التى أدت إلى تجزئة الجدول، فإن البديل المقبول الوحيد هو البديل الثالث: وذلك لأن بقية البدائل سـينتج عنها ما يسـمى «السجلات الزائفة» (Spurious Tuples) عنـد إجرائنا لعمليـة «ربط» (Join) بين الجداول الناتجة مـن كل بديل للحصول على البيانـات الأصليـة الموجودة فى الجدول الأصلى قبل تجزئته. فعلى سـبيل المثال، لو اسـتخدمنا البديل الأول فى تجزئة جدول «الموظف - المشروع» سينتج عنه الجدولان التاليان:

EMP_PROJ2

ENO	PNO	Duration	Responsibility
E1	P1	12	Manager
E2	P3	20	System Analyst
E2	P2	10	System Analyst
E3	P3	10	Consultant
E3	P4	14	Consultant
E4	P2	26	Programmer
E5	P2	14	Manager
E6	P4	16	Programmer
E7	P3	12	Manager
E8	P3	12	Project Secretary

EMP_PROJ1

ENO	PLocation
E1	Riyadh
E2	Jizan
E2	Jouf
E3	Joubail
E3	Skaka
E4	Jouf
E5	Jouf
E6	Hail
E7	Jizan
E8	Jizan

وعند إجرائنا لعملية ربط (طبيعى) بين الجدولين الناتجين من البديل الأول باستخدام الحقل المشترك بين الجدولين وهو حقل «رقم الموظف» (ENO)، سينتج أربعة سجلات زائفة لكونها غير موجودة ضمن سجلات الجدول الأصلى (قبل عملية التجزئة). كذلك هو الحال لو استخدمنا البديل الثانى حيث سينتج عن عملية ربط جدولى البديل الثانى سجلات زائفة. ويوضح الجدول التالى السجلات الزائفة الناتجة من عملية ربط جدولى البديل الأول موضحة من خلال وضع علامة النجمة (*) أمامها.

EMP_PROJ1_Join_ EMP_PROJ2

ENO	PLocation	PNO	Duration	Responsibility	
E1	Riyadh	P1	12	Manager	
E2	Jizan	P3	20	System Analyst	
E2	Jizan	P2	10	System Analyst	*
E2	Jouf	P3	20	System Analyst	*
E2	Jouf	P2	10	System Analyst	
E3	Joubail	P3	10	Consultant	
E3	Joubail	P4	14	Consultant	*
E3	Skaka	P3	10	Consultant	*
E3	Skaka	P4	14	Consultant	
E4	Jouf	P2	26	Programmer	
E5	Jouf	P2	14	Manager	
E6	Hail	P4	16	Programmer	
E7	Jizan	P3	12	Manager	
E8	Jizan	P3	12	Project Secretary	

ونظراً لأن الشكل الطبيعى الثالث والشكل الطبيعى «بويس - كود» ينصان على أن شروطهما يجب أن تنطبق على الاعتماديات الوظيفية كافة مما يعنى أنه يجب أن تنطبق شروطهما على الاعتماديات الوظيفية الضمنية التى يمكن أن يستدل عليها (أو تستنبط)، بالإضافة إلى الظاهرة منها، فإن هذا يدعونا إلى التعرف على قواعد الاستدلال. كما أن قواعد الاستدلال هذه ضرورية لمعرفة إن كان أى تجزىء لجدول ما يحافظ على خاصية «المحافظة على الاعتماديات الوظيفة» (Dependency Preservation) وخاصية «السجلات غير الزائفة» (Lossless Decomposition).

٦-١-٤ قواعد الاستدلال (Inference Rules):

يقوم مصممو قواعد البيانات عادة بتعريف الاعتماديات الوظيفية المتعلقة بكل جدول. وتكون هذه الاعتماديات الوظيفية ذات معانٍ واضحة، ويجب أن تتحقق فى

أى حالة يكون عليها الجدول، كما رأينا فى الأمثلة أعلاه. ويستخدم عادة الرمز "F" للدلالة على هذه الاعتماديات الوظيفية واختصاراً لعبارة «اعتماديات وظيفية» (Functional Dependencies). إلا أنه يوجد عادة الكثير من الاعتماديات الوظيفية التى تنطبق على كل حالة من حالات الجدول غير تلك المعرفة فى "F". ويمكن الاستدلال على هذه الاعتماديات الوظيفية الأخرى من خلال الاعتماديات الوظيفية المعرفة فى "F": إذ إنه يصعب، بشكل عام، تعريف جميع الاعتماديات الوظيفية لتمثيل حالة معينة. فعلى سبيل المثال، لنفترض أن «رقم عضو هيئة التدريس» يحدد «رقم القسم» الذى يعمل فيه عضو هيئة التدريس (Faculty_No → Dept_No) وأن «رقم القسم» يحدد «اسم القسم» (Dept_No → Dept_Name). بالنظر فى هاتين الاعتماديتن، معاً، نستدل أن «رقم عضو هيئة التدريس» يحدد «اسم القسم» الذى يعمل فيه (Faculty_No → Dept_Name). وتعد الاعتمادية الثالثة، فى هذه الحالة، اعتمادية وظيفية يمكن الاستدلال عليها من خلال الاعتماديتن الوظيفيتين الأخريين، ولا داعى لإدراجها بالإضافة إلى الاعتماديتين الوظيفيتين الأخريين ضمن الاعتماديات الوظيفية الواجب إيضاحها. وبشكل رسمى، يمكن تعريف الاعتماديات الوظيفية كافة التى يمكن أن يستدل عليها من خلال مجموعة الاعتماديات الوظيفية الواضحة "F" فيما يعرف بمبدأ «الانغلاق» (Closure)، كما يلى:

إن مجموعة الاعتماديات الوظيفية التى تحتوى على مجموعة الاعتماديات الوظيفية الواضحة "F" بالإضافة إلى جميع الاعتماديات الوظيفية التى يمكن أن يستدل عليها من "F" تسمى انغلاق "F" (Closure of F) ويرمز لها بالرمز "F^+".

وعلى سبيل المثال، لنفترض تعريف الاعتماديات الوظيفية الواضحة "F" كما يلى:

F = {Faculty_No → {Name, Salary, DOB, Dept_No},
Dept_No → {DName, DLocation}}

فإنه يمكن الاستدلال على اعتماديات وظيفية أخرى من الاعتماديات الوظيفية الواضحة "F"، من ضمنها ما يلى:

Faculty_No → {DName, DLocation}
Faculty_No → Faculty_No
Dept_No → DName

وعندما يستدل على اعتمادية وظيفية فإن هذه الاعتمادية الوظيفية يجب أن تتحقق على حالات الجدول كافة، كما هو الحال بالنسبة للاعتماديات الوظيفية الواضحة.

وللتعــرف على الاعتماديات الوظيفية كافة التى يمكن أن يســتدل عليها من مجموعة من الاعتماديات الوظيفية الواضحة بشــكل نمطى فإننا بحاجة إلى قواعد اســتدلال تمكننا من ذلك. ومن أهم قواعد الاستدلال قواعد استدلال «أرمسترونغ» التى تسمى عادة «حقائق أرمســترونغ» (Armstrong's Axioms) أو «قواعد اســتدلال أرمســترونغ» (Armstrong's Inference Rules)، وهى كما يلى:

١- الازدياد (Augmentation): $\{X \rightarrow Y\} \Rightarrow \{XZ \rightarrow YZ\}$

٢- الانتقال (Transitivity): $\{X \rightarrow Y, Y \rightarrow Z\} \Rightarrow \{X \rightarrow Z\}$

٣- الانحسار (Reflexivity): $W \subseteq X \Rightarrow \{X \rightarrow W\}$

تنص القاعدة الأولى أنه بإضافة أى مجموعة من الحقول إلى جانبى أى اعتمادية وظيفيــة تنتج اعتمادية وظيفية صحيحة جديــدة. أما القاعدة الثانية فتنص على أن الاعتماديات الوظيفية انتقالية. القاعدة الثالثة تنص على أن أية مجموعة من الحقول تحدد أية مجموعة جزئية من الحقول نفســها. وتعد حقائق أرمســترونغ «ســليمة» (Sound) بمعنى أنه لا يمكن أن ينتج عنها أية اعتماديات وظيفية خاطئة ليست فى "F^+". كما أن حقائق أرمســترونغ تعد «كاملة» (Complete)، بمعنى أنه يمكن لأية مجموعة من الاعتماديات الوظيفية الواضحة "F" الوصول إلى الاعتماديات الوظيفية كافةً فى "F^+" باستخدام هذه القواعد الثلاث فقط.

وعلى الرغم من أن قواعد أرمسترونغ كاملة، إلا أنه قد تستخدم القواعد الإضافية التالية لتفهم بعض الاعتماديات فى "F^+". وهذه القواعد الإضافية كما يلى:

١- الاتحاد (Union): $\{X \rightarrow Y, X \rightarrow Z\} \Rightarrow \{X \rightarrow YZ\}$

٢- التفكيك (Decomposition): $\{X \rightarrow YZ\} \Rightarrow \{X \rightarrow Y, X \rightarrow Z\}$

٣- الانتقال الزائف (Pseudotransitivity): $\{X \rightarrow Y, YW \rightarrow Z\} \Rightarrow \{XW \rightarrow Z\}$

ويمكن الاستدلال على جميع الاعتماديات الوظيفية "F^+" من الاعتماديات الوظيفية الواضحة "F" بشــكل مبســط إذا تم النظر إلى المســألة على أســاس معرفة انغلاق **مجموعــة من الحقــول** (X^+) وليس معرفة انغلاق **مجموعة من الاعتماديات** الوظيفية (F^+). وتعبــر الدالة (Function) التالية عــن خوارزمية يمكن من خلالها معرفة انغلاق أى حقــل (أو حقول) بحيث تكون مدخلات هذه الدالة مجموعة الحقول المراد معرفة انغلاقهــا، والممثلة فــى (X) ومجموعة الاعتماديات الوظيفيــة المنطبقة على الجدول والممثلة فى (F).

```
function ComputeX+ (X, F)
begin
X+ ← X
While there exists Y → Z ∈ F such that
        Y ⊆ X+ and Z ⊈ X+
        then X+ ← X+ ∪ Z
return (X+)
end
```

تبدأ الدالة السـابقة بوضع جميع الحقول المراد معرفة انغلاقها (X) ضمن انغلاق الحقول (X^+) نفسها، وذلك حسب قاعد الانحسار، وذلك لأن أية مجموعة من الحقول تحـدد قيم أية مجموعة جزئية من الحقول نفسـها. بعد ذلك ينظر فى كل اعتمادية وظيفية مطبقة على الجدول (واحدة تلو الأخرى). ولكل اعتمادية وظيفية ينظر فيما إذا كانـت هذه الاعتمادية تمثـل اعتمادية انتقالية على الحقـول التى تمت معرفتها باعتبارهـا جـزءاً من انغلاق (X) وهـو (X^+). فإذا كانت هـذه الاعتمادية هى بالفعل انتقالية وأن حقول جانبها الأيمن لم تدرج ضمن (X^+) بعد، تسـتخدم قاعدة الانتقال، بحيـث تتم إضافة هذه الحقول ضمن (X^+). ومثال تطبيقى على هذه الدالة لنفترض وجود جدول (R) تنطبق عليه الاعتماديات الوظيفية الظاهرة (F) التالية.

$$F = \{A \rightarrow B, C \rightarrow \{D, E\}, \{E, G\} \rightarrow H\}$$

ولنفترض أننا نرغب فى معرفة انغلاق الحقلين C و G. فى هذه الحالة تتم مناداة الدالة كما يلى:

$$ComputeX^+ (\{C, G\}, F)$$

وبناءً على هذه المعطيات، يكون عمل الدالة كما يلى:

- **الوضع المبدئى:** $X^+ = \{C, G\}$

- **الدورة الأولى:** ينظر فى الاعتمادية الوظيفية التى يكون جزؤها الأيسر مجموعة مـن الحقـول التى تدخل ضمـن X^+. ينظر - إذَنْ - فى هذه الحالـة إلى الاعتمادية الوظيفية الثانية ($C \rightarrow \{D,E\}$) لكون جانبها الأيسـر مـن ضمن الحقول الموجودة فى X^+ وحقولها اليمنى ليسـت من ضمن حقول X^+. ونظراً لأن حقول الجانب الأيمن فى

الاعتمادية ليســت من ضمن حقول X^+ على الرغم من أن الحقل الذى يحدد الجانب الأيمن من ضمن X^+، وهو (C)، تتم إضافة الحقول اليمنى من الاعتمادية لتصبح جزءاً من X^+. وبذلك تكون نتيجة هذه الدورة كما يلى: $X^+ = \{C, G, D, E\}$.

- **الــدورة الثانيــة:** ينظر مرة أخــرى فى الاعتمادية الوظيفيــة التى يكون جزؤها الأيســر مجموعة من الحقول التــى تدخل ضمن X^+. إذاً ينظر هنــا إلى الاعتمادية الثالثة ($\{E, G\} \rightarrow H$) لكون جانبها الأيسر من ضمن الحقول الموجودة فى X^+ وحقولها اليمنــى ليس من ضمن حقــول X^+. ونظراً لأن حقل الجانب الأيمــن فى الاعتمادية ليســت من ضمن حقول X^+ على الرغم من أن الحقول التى تحدد الجانب الأيمن هى من ضمن X^+، تتم إضافة الحقل الأيمن من الاعتمادية ليصبح جزءاً من X^+. ويلاحظ فى هذه الحالة أنه تم اســتخدام قاعدة الانتقــال ضمن خوارزمية الدالة عند تحديد الجانب الأيمن من الاعتمادية الثالثة. وبذلك تكون نتيجة هذه الدورة كما يلى: $X^+ = \{C, G, D, E, H\}$.

- **التوقف:** يتم التوقف بعد الدورة الثانية لعدم وجود أية اعتماديات وظيفية جانبها الأيسر من ضمن الحقول التى تم التعرف عليها وتدخل ضمن حقول X^+.

وإذا طبقنا هذه الدالة على جدول «الموظف - المشــروع» ذى الاعتماديات الوظيفية التالية:

F = {ENO → {ENmae, Title, Salary},
PNO → {PName, Budget},
{ENO, PNO} → {PLocation, Duration, Responsibility},
Title → Salary,
PLocation → PNO}

وذلك بغية معرفة انغلاق بعض حقول الجدول، فســيكون انغلاق هذه الحقول كما يلى:

1- $\{ENO\}^+$ = {ENO, EName, Title, Salary}
2- $\{PNO\}^+$ = {PNO, PName, Budget}
3- $\{ENO, PNO\}^+$ = {ENO, PNO, EName, Title, Salary, PName, Budget, PLocation, Duration, Responsibility}
4- $\{Title\}^+$ = {Title, Salary}
5- $\{PLocation\}^+$ = {PLocation, PNO, PName, Budget}
6- $\{ENO, PLocation\}^+$ = {ENO, PNO, EName, Title, Salary, PName, Budget, PLocation, Duration, Responsibility}

ويمكن تعديل الدالة أعلاه بحيث تستخدم للتعرف على المفاتيح المرشحة للجدول. ويتم ذلك من خلال معرفة انغلاق كل حقل من حقول الجدول على حدة، فإذا كانت نتيجة الدالة جميع حقول الجدول يعنى أن الحقل مفتاح مرشح. بعد ذلك يتم معرفة انغلاق كل حقلين من حقول الجدول فإذا كانت نتيجة الدالة لأى حقلين جميع حقول الجدول ولم يوجد حقل منها سبق أن كان مفتاحاً مرشحاً للجدول فى الخطوة الأولى، يكون الحقلان مفتاحاً مرشحاً للجدول. وتستمر العملية وصولاً إلى حقول الجدول كافة. وتفيد هذه العملية لكون الشكل الطبيعى الثالث والشكل الطبيعى «بويس - كود» ينصان، ضمن شروطهما، على أن يكون الجانب الأيسر من أية اعتمادية وظيفية مفروضة على جدول ما مفتاحاً خارقاً للجدول. ويعنى هذا أن الجانب الأيسر يحتوى ضمن حقوله على مفتاح مرشح.

٦-١-٥ خواص التجزئة (Properties of Decomposition):

يعتمد تطبيع الجداول على مبدأ التجزئة للحد من تكرار البيانات غير المرغوب فيها. وكما أوضحنا عند شرح الشكل الطبيعى «بويس - كود»، قد ينتج عن عمليات التجزئة ظهور بعض المشكلات التى لم تكن موجودة أصلاً. وبشكل خاص، يجب أن نتأكد من أن تجزئة أى جدول تمكننا من استرجاع البيانات الأصلية فى الجدول الأصلى عند إجراء عملية ربط بين الجداول الناتجة من عملية التجزئة، كما يجب أن نتأكد من أن تجزئة الجدول تمكننا من التحقق من انطباق قيود التكامل، بعد التجزئة، بشكل فعال. وفيما يلى شرح للخاصيتين اللتين تمكناننا من اختبار ذلك.

٦-١-٥-١ خاصية المحافظة على الاعتماديات الوظيفية (Dependency Preservation Decomposition):

إنه من المفيد إذا وجدت كل اعتمادية وظيفية ($X \rightarrow Y$)، موجودة أصلاً ضمن الاعتماديات الوظيفية المفروضة على جدول، وهى "F"، ضمن أحد الجداول الناتجة من عملية تجزئة الجدول أو كان بالإمكان الاستدلال عليها من حقول أحد الجداول بعد التجزئة. وتسمى هذه الخاصية «المحافظة على الاعتماديات الوظيفية». وتظهر الحاجة إلى هذه الخاصية لكون كل اعتمادية وظيفية تمثل قيداً بين حقول الجدول الأصلى يجب التأكد من انطباقها على حالات الجدول كافة. وإذا لم توجد إحدى الاعتماديات الوظيفية ضمن أحد الجداول الناتجة من عملية التجزئة فإننا لن نتمكن

من فرض هذه الاعتمادية من خلال التعامل مع جدول واحد، ولكنه يجب إجراء عملية ربــط بين جدولين أو أكثر، ومن ثم التحقق من انطبــاق الاعتمادية على نتيجة عملية الربط. ولكــون عملية الربط عملية تتطلب بعض الوقت لتنفيذها، فإن هذه الطريقة تعــد غير فعالة وغير عملية. وتجدر الإشــارة إلى أنه ليس مــن الضرورى أن توجد الاعتماديــات الوظيفيــة الموجودة فى "F" ذاتها، كل على حــدة، ضمن أحد الجداول الناتجة من عملية التجزئة، لكنه يكفى أن يكون اتحاد الاعتماديات الوظيفية الموجودة فــى الجداول، كل على حدة، مكافئــة للاعتماديات الوظيفية الموجودة فى "F". وفيما يلى التعريف الرسمى لخاصية «المحافظة على الاعتماديات الوظيفية»:

> تعــد تجزئة هيــكل الجدول "R" إلى أكثــر من جــدول $(R_1, R_2, \ldots, R_n)$ تجزئة تحافظ علــى الاعتماديات الوظيفية إذا كانــت كل اعتمادية وظيفية فى "F"، بما فى ذلك التى يمكن الاســتدلال عليها، متحققة فى اتحاد عمليات إسقاط الاعتماديات الوظيفية على الجداول الناتجة من عملية التجزئة، كما يلى:
>
> $$((\pi_{R_1}(F)) \cup (\pi_{R_2}(F)) \cup \ldots \cup (\pi_{R_n}(F)))^+ = F^+$$

ومثالاً على فقد الاعتماديات الوظيفية، لننظر فى مثال «الموظف – المشــروع» الذى تمت تجزئته عند شــرحنا للشكل الطبيعى «بويس – كود» ونتج عنه ثلاثة بدائل للتجزئة وهى:

١- (ENO, PNO, Duration, Responsibility) و (ENO, PLocation)

٢- (PNO, ENO, Duration, Responsibility) و (PNO, PLocation)

٣- (PLocation, ENO, Duration, Responsibility) و (PLocation, PNO)

وبإجراء عمليات إســقاط، حسب التعريف أعلاه، للاعتماديات الوظيفية الظاهرة على الجداول الناتجة من البدائل الثلاثة، تكون نتيجة عملية الإسقاط، موضحة حسب كل بديل، كما يلى:

١- $(\{ENO, PNO\} \rightarrow \{Duration, Responsibility\})$

٢- $((PLocation \rightarrow PNO), (\{PNO, ENO\} \rightarrow \{Duration, Responsibility\}))$

٣- $((PLocation \rightarrow PNO), (\{PLocation, ENO\} \rightarrow \{Duration, Responsibility\}))$

فى البديل الأول تم فقد الاعتمادية الوظيفية (PLocation → PNO) وكذلك الاعتمادية الوظيفيـــة ({PNO, ENO} → {PLocation}). وحتـــى لو تم ربـــط الجدولين الناتجين من عملية التجزئة فى هذا البديل فإننا لن نتمكن من فرض هاتين الاعتماديتين الوظيفيتين؛ وذلك لأن هذا البديل يعانى مشـــكلة أكبر وهى وجود ســـجلات زائفة (Spurious Tuples) كمـــا أوضحنا أعلاه. أما البديل الثانى فإنه يؤدى أيضـــاً إلى فقد اعتماديات وظيفية، إلا أنـــه يفقد اعتمادية وظيفية واحدة، وليس اعتماديتين اثنتين كما هو الحال بالنســـبة للبديـــل الأول. وهذه الاعتمادية هـــى ({PNO, ENO} → {PLocation}). وكما هو الحال بالنســـبة للبديل الأول، فإن هذا البديل يعانى مشـــكلة أكبر لـــن تمكننا من فرض هذه الاعتمادية؛ لأنه ســـينتج عن عملية الربط بين جدولى هذا البديل ســـجلات زائفة. أما البديل الثالث فسيفقد الاعتمادية الوظيفية ({PNO, ENO} → {PLocation}). إلا أن هذا البديل لن ينتج عنه سجلات زائفة فى أثناء عملية ربط جدولية، ومن ثم يمكننا التحقق من انطباق الاعتمادية الوظيفية التى فقدت فى أثناء التجزئة. ويعنى هذا أن كل البدائل الثلاثـــة لتجزئة الجدول تؤدى إلى فقد اعتمادية وظيفية أو أكثر مما يجب إجراء عملية ربـــط بين الجداول الناتجة للتحقق من انطباق الاعتماديات الوظيفية التى فقدت نتيجة لعمليـــة التجزئة، وما دامت عملية التجزئة لا ينتج عنها وجود ســـجلات زائفة فى أثناء ربط الجداول الناتجة من عملية التجزئة. ولهذا الســـبب فإن خاصية «الســـجلات غير الزائفة» تعد أهم بكثير من خاصية «المحافظة على الاعتماديات الوظيفية»، لكون الأولى تعنـــى بصحة البيانات الناتجة بعد عملية الربط، على حين أن الثانية تعنى بأداء النظام الذى قد يتأثر نتيجة عمليات الربط بغية التحقق من انطباق الاعتماديات الوظيفية التى فقدت نتيجة لعملية التجزئة.

٦-١-٥-٢ خاصية السجلات غير الزائفة (Lossless Join Decomposition):

إن الخاصية الثانية التى يجب لأى تجزئة لجدول أن تتحلى بها هى خاصية السجلات غير الزائفة. وكما أوضحنا عند شرحنا للشكل الطبيعى «بويس – كود»، أنه من الممكن أن تتم تجزئة الجدول وفق أكثر من بديل، ولكن بعض هذه البدائل قد يؤدى إلى إضافة ســـجلات زائفة عند إجراء عملية ربط بين الجداول الناتجة من التجزئة بغية استعادة البيانات الموجودة فى الجدول الأصلى قبل تجزئته. ويعنى هذا أن الســـجلات الزائفة التى قد تتم إضافتها بعد عملية الربط تمثل بيانات خاطئة لكونها غير موجودة أصلاً ضمن حالات الجدول قبل تجزئته. وفيما يلى التعريف الرسمى لهذه الخاصية:

تعد تجزئة هيكل الجدول "R" إلى أكثر من جدول $(R_1, R_2, \ldots, R_n)$ تجزئة لا تؤدى إلى وجود سجلات زائفة عند ربط الجداول الناتجة من عملية التجزئة ربطاً طبيعياً بالنسبة لمجموعة الاعتماديات الوظيفية "F" المفروضة على R إذا انطبق الشرط التالى على أية حالة "r" من الحالات التى قد يكون عليها الجدول R وتنطبق عليها جميع الاعتماديات الوظيفية المفروضة عليه "F":

$$(\pi_{R_1}(r) \bowtie (\pi_{R_2}(r) \bowtie \ldots \bowtie \pi_{R_n}(r)) = r$$

ويعنـــى التعريـــف أعلاه أنه لو أخذنا أية حالة "r" يكـــون عليها الجدول R وأجرينا عملية إســـقاط لحقول الجدول وفق حقول الجداول التـــى تمثل تجزئة الجدول، فإنه يجـــب أن تكون نتيجة عملية الربط الطبيعى بـــين جميع جداول التجزئة حالة مكافئة لحالـــة الجدول الأصلى وهـــى "r". وبمعنى آخر، يجب أن تكـــون نتيجة ربط جداول التجزئـــة تحتوى على بيانات تكافـــئ البيانات الموجودة فى الجدول الأصلى دون وجود أى سجلات إضافية.

وللتأكد من أن تجزئة أى جدول (R) إلى جدولين (R_1, R_2) تتحلى بخاصية السجلات غير الزائفة يمكن إجراء الاختبار التالى:

إن تجزئة أى جدول (R) إلى جدولين (R_1, R_2) تتحلى بخاصية السجلات غير الزائفة بالنسبة للاعتماديات الوظيفية "F" المفروضة على (R) إذا تحقق أى من الشرطين التاليين:

- توجد اعتمادية وظيفية فى "F^+" على الشكل التالى: $(R_1 \cap R_2) \rightarrow (R_1 - R_2)$
- توجد اعتمادية وظيفية فى "F^+" على الشكل التالى: $(R_1 \cap R_2) \rightarrow (R_2 - R_1)$

ويعنى الاختبار الســـابق أنه عندما نقوم بتجزئة جدول إلى جدولين للحصول على شـــكل طبيعى ما، فإن الجدولين الناتجين يتحليان بخاصية السجلات غير الزائفة إذا كانت الحقول المشتركة بين الجدولين تحدد جميع حقول الجدول الأول أو جميع حقول الجدول الثانى. وبمعنى آخر، يجب أن تكون الحقول المشتركة بين الجدولين الناتجين مفتاحاً رئيسياً للجدول الأول أو مفتاحاً رئيسياً للجدول الثانى. وبناءً على ذلك، يمكن إعادة تعريف الشرطين كما يلى:

- $(R_1 \cap R_2) \rightarrow R_1$
- $(R_1 \cap R_2) \rightarrow R_2$

وبنـاءً علـى الاختبار هذا، يمكـن التحقق مـن أن البديل الثالـث لتجزئة جدول «الموظف - المشـروع» أعلاه هو البديل الوحيد الذى يتحلى بخاصية السـجلات غير الزائفة. ففى هذا البديل يوجد حقل واحد مشـترك بين الجدولين وهو حقل «موقع المشـروع» (PLocation)، وهذا الحقل يمثل المفتاح الرئيسـى لأحد الجدولين. أما فى البديل الأول فيوجد أيضاً حقل مشترك واحد بين جدولى هذا البديل وهو حقل «رقم الموظف» (ENO)، ولكن هذا الحقل ليس مفتاحاً رئيسـياً لأى من الجدولين. كذلك هو الحـال فى البديل الثانى الذى يوجد فيه حقل مشـترك واحد بين جدوليه، وهو حقل «رقم المشروع» (PNO)؛ ولكنه ليس مفتاحاً رئيسياً لأى من جدولى هذا البديل.

٦-١-٥-٣ التجزئة التى تتحلى بخاصية «المحافظة على الاعتماديات الوظيفية» (Dependency Preservation) وخاصية «السجلات غير الزائفة» (Lossless Join Decomposition):

لتطبيع جدول حتى يكون فى الشـكل الطبيعى الثالث فإنه من الممكن دائماً تجزئته إلـى جدولين (أو أكثر)، على أن تتحلى تجزئته بكل من خاصية الاعتماديات الوظيفية وخاصية السـجلات غيـر الزائفة. وتمثل الخوارزمية التاليـة، تعميماً للطريقة التى اسـتخدمناها عند شـرح الشـكل الطبيعى الثالث لتطبيع أى جدول ليصبح بالشكل الطبيعى الثالث مع المحافظة على كلتا الخاصيتين.

المدخـلات: جـدول ليس فى الشـكل الطبيعى الثالـث، والاعتماديات الوظيفية المنطبقة عليه.

المخرجات: جداول فى الشكل الطبيعى الثالث تتحلى بخاصية «المحافظة على الاعتماديات الوظيفية» وخاصية «السجلات غير الزائفة».

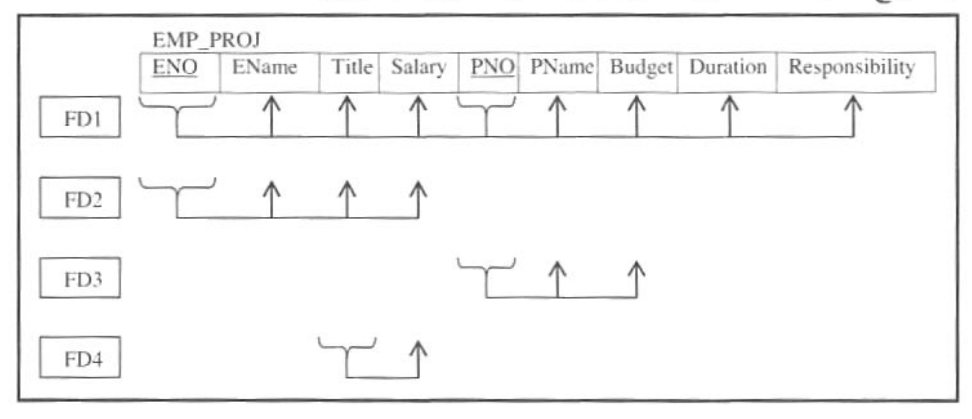
الخطوات: لكل الحقول الموجودة فى الجانب الأيمن من أية اعتمادية وظيفية تحددها نفس حقول الجانب الأيسر، وتكون الاعتمادية الوظيفية مخالفة لشروط الشكل الطبيعى الثالث:

١- يتم إنشاء جدول جديد يحتوى على كل الحقول الموجودة فى الجانب الأيمن التى تحددها نفس حقول الجانب الأيسر بالإضافة إلى حقول الجانب الأيسر.

٢- يكون مفتاح الجدول الجديد هو الحقول الموجودة فى الجانب الأيسر.

٣- تُزَال الحقول التى فى الجانب الأيمن من الجدول الأصلى الذى هو قيد التجزئة.

ولإيضــاح طريقة عمل الخوارزمية الســابقة، نعود مرة أخرى إلى مثال «الموظف – المشروع» المبينة اعتمادياته الوظيفية، مرة أخرى، كما يلى:



وكما ســبق أن أوضحنا، فإن الاعتمادية الوظيفية الثانية، والثالثة، والرابعة تخالف شروط الشكل الطبيعى الثالث لكونها ليست اعتماديات وظيفية بديهية، وليس الجانب الأيمن فيها حقولاً أولية، كما أن الجانب الأيسر فى كل منها ليس مفتاحاً خارقاً. وبناءً على ذلك يتم إنشــاء جدول جديد لكل واحدة من هذه الاعتماديات المخالفة لشــروط الشــكل الطبيعى الثالث حيث يتم إنشــاء جدول يحتوى على جميع الحقول الممثلة فى الاعتماديــة الثانيــة، ويكون مفتاح هذا الجدول الحقل الممثل فى الجانب الأيســر من الاعتماديــة وهو حقل «رقم الموظف» (ENO). وتتم إزالــة الحقول الممثلة فى الجانب الأيمــن من الاعتمادية مــن حقول الجدول الأصلى. وتكون نتيجــة هذه التجزئة كما يلى:

1- EMP_PROJ (ENO, PNO, PName, Budget, Duration, Responsibility)

2- EMPLOYEE (ENO, EName, Title, Salary)

وينشــأ جدول جديــد ثانٍ لتمثيل الاعتمادية الوظيفيــة الثالثة بحيث تكون حقول هذا الجدول مطابقة لتلك الممثلة فى الاعتمادية. وكما هو الحال فى الجدول الجديد الذى تم إنشاؤه سابقاً، يكون المفتاح الرئيسى للجدول قيد الإنشاء هو الحقل الممثل فى الجانب الأيســر من الاعتمادية، وهو حقل «رقم المشــروع» (PNO). وتتم إزالة الحقول الممثلــة فى الجانب الأيمن من الاعتمادية من حقــول الجدول الأصلى. وتكون نتيجة التجزئة حتى هذه المرحلة كما يلى:

1- EMP_PROJ (ENO, PNO, Duration, Responsibility)

2- EMPLOYEE (ENO, EName, Title, Salary)

3- PROJECT (PNO, PName, Budget)

وبذلـــك يصبح كل من الجدول الأصلى (EMP_PROJ) والجدول الثالث (PROJECT) فى الشـــكل الطبيعى الثالـــث، إلا أن الجدول الثانى (EMPLOYEE) ليس فى الشـــكل الطبيعى الثالث لوجود الاعتمادية الوظيفية الرابعة فيه. لذا يتم إنشـــاء جدول جديد تكون حقولـــه تلك الممثلة فى الإعتماديـــة الوظيفية الرابعة، ويكون المفتاح الرئيســـى للجـــدول الجديد هـــو الحقل الموجود فى الجهة اليســـرى مـــن الاعتمادية وهو حقل «مســـمى الوظيفة» (Title). وتتم إزالة حقـــل الجانب الأيمن فى الاعتمادية من حقول جـــدول الموظفين (EMPLOYEE). وبذلك تكون التجزئة النهائية للجدول أربعة جداول كل منها فى الشكل الطبيعى الثالث، كما يلى:

1- EMP_PROJ (ENO, PNO, Duration, Responsibility)

2- EMPLOYEE (ENO, EName, Title)

3- PAYMENT (Title, Salary)

4- PROJECT (PNO, PName, Budget)

٦-١-٥-٤ التجزئة إلى الشكل الطبيعى بويس - كود مع المحافظة على خاصية «السجلات غير الزائفة» (Lossless Join Decomposition):

ســـبق أن أشـــرنا أعلاه إلى أن هناك تجزئة لأى جدول ليس فى الشـــكل الطبيعى الثالث، بحيث تكون نتيجة تجزئته عبارة عن جداول بالشـــكل الطبيعى الثالث، وتنطبق على هـــذه التجزئة كل من خاصية «المحافظة على الاعتماديـــات الوظيفية» وخاصية «الســـجلات غير الزائفة». إلا أن هذا من غير الممكن دائماً عند تطبيع جدول ليصبح بالشـــكل الطبيعى «بويس - كود» بمعنى أن الجداول الناتجة من عملية التجزئة قد لا تنطبق عليها خاصية «المحافظة على الاعتماديات الوظيفية»، كما سبق أن أوضحنا فى مثال «الموظف - المشـــروع» الذى كانت جميع بدائل تجزئته الثلاث لا تتحلى بخاصية «المحافظة على الاعتماديات الوظيفية». ولأن خاصية «الســـجلات غير الزائفة» أهم كثيراً من خاصية «المحافظة علـــى الاعتماديات الوظيفية»، ونظراً لدوام وجود تجزئة لجدول حتى يصبح بالشكل الطبيعى «بويس - كود» مع تحليه بخاصية «السجلات غير

الزائفة»، فإننا بحاجة إلى طريقة تمكننا من الوصول إلى هذه التجزئة بشــكل فعال. وتمثل الخوارزمية التالية هذه الطريقة:

المدخلات: جدول (R) ليس فى الشــكل الطبيعى «بويس - كود»، والاعتماديات الوظيفية المنطبقة عليه (F).

المخرجات: مجموعــة جداول (D) فى الشــكل الطبيعى «بويــس - كود» تتحلى بخاصية «السجلات غير الزائفة».

الخطوات:

١- تتكون التجزئة (D) مبدئياً من جدول واحد هو (R)

$$D \leftarrow (R)$$

٢- مادام وجد جدول ضمن جداول التجزئة لا تنطبق عليه شروط الشكل الطبيعى «بويس - كود» يتم عمل التالى:

- إذا كانت الاعتمادية الوظيفية المخالفة للشكل الطبيعى بويس كود هى:

$$X \rightarrow Y$$

تتم التجزئة كما يلى:

$$D \leftarrow (D - Q) \cup (Q - Y) \cup (X \cup Y)$$

ومثال على طريقة عمل الخوارزمية السابقة، نعود مرة أخرى إلى جدول «الموظف - المشروع» التالى:

EMP_PROJ (ENO, PNO, EName, Title, Salary, PName, Budget, PLocation, Duration, Responsibility)

الذى تنطبق عليه خمس اعتماديات وظيفية وهى كالتالى:

FD1: {ENO, PNO} → {EName, Title, Salary, PName, Budget, PLocation, Duration, Responsibility}

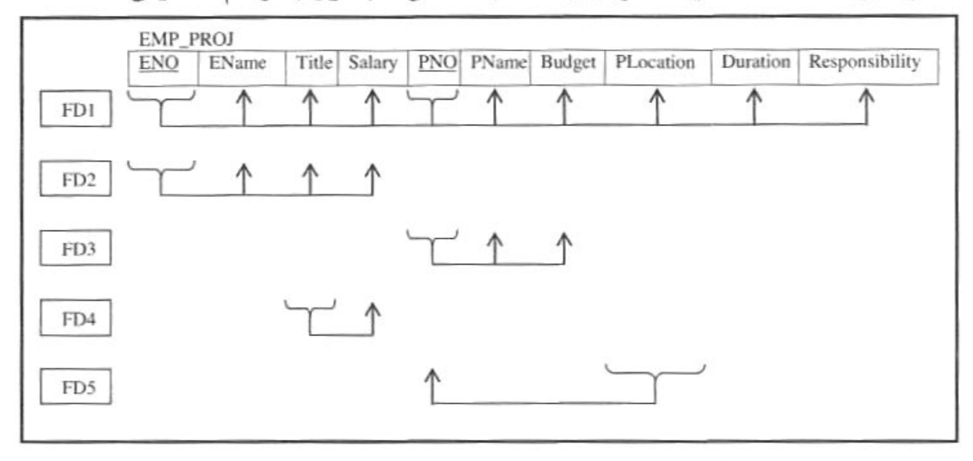
FD2: ENO → {EName, Title, Salary}

FD3: PNO → {PName, Budget}

FD4: Title → Salary

FD5: PLocation → PNO

وتمثل هذه الاعتماديات الوظيفية المطبقة على الجدول بالرسم كما يلى:



وكما أسلفنا سابقاً، إن جدول «الموظف - المشروع» أعلاه ليس فى الشكل الطبيعى «بويس - كود»، وذلك بسبب الاعتماديات الوظيفية الثانية والثالثة والرابعة والخامسة؛ إذ إن الجانب الأيسر فى كل منها ليس مفتاحاً خارقاً للجدول. ولتطبيع الجدول ليصبح فى الشكل الطبيعى «بويس - كود» نستخدم الخوارزمية السابقة، بافتراض أن اسم الجدول هو (R_1)، كما يلى:

- **الوضع المبدئى:** تتكون التجزئة من جدول واحد حقوله هى جميع حقول الجدول الرئيسى: $D = \{R_1\}$

- **الدورة الأولى:** R_1 الموجود ضمن جداول التجزئة ليس فى الشكل الطبيعى «بويس - كود». يتم اختيار إحدى الاعتماديات الوظيفية المنطبقة على R_1، وفى الوقت نفسه مخالفة للشكل الطبيعى «بويس - كود»، ولتكن:

$$ENO \rightarrow \{EName, Title, Salary\}$$

تطبق قاعدة التجزئة $D \leftarrow (D - Q) \cup (Q - Y) \cup (X \cup Y)$. ولأن (D) فى الدورة الأولى تحتوى على جدول واحد فهو أيضاً (Q). وتكون نتيجة تطبيق القاعدة كما يلى:

$$D \leftarrow \emptyset \cup R_2 \cup R_3$$

بحيث إن:

R_2(<u>ENO</u>, <u>PNO</u>, PName, Budget, PLocation, Duration, Responsibility)

R_3 (<u>ENO</u>, EName, Title, Salary)

- **الدورة الثانية:** تكون جداول التجزئة حتى الآن كما يلى $D = \{R_2, R_3\}$

الجدول R_2 والجدول R_3 ليسا فى الشكل الطبيعى «بويس - كود». تتم تجزئة الجدول R_3 وفق الاعتمادية الوظيفية المنطبقة على R_3 وفى الوقت نفسه مخالفة للشكل الطبيعى «بويس - كود»، وهى:

$$Title \rightarrow Salary$$

تطبق قاعدة التجزئة $D \leftarrow (D - Q) \cup (Q - Y) \cup (X \cup Y)$ على الجدول R_3، كما يلى:

$$D \leftarrow (R_2) \cup (R_4) \cup (R_5)$$

بحيث إن:

R_4 (<u>ENO</u>, EName, Title)

R_5 (<u>Title</u>, Salary)

- **الدورة الثالثة:** تكون جداول التجزئة حتى الآن كما يلى $D = \{R_2, R_4, R_5\}$

الجدول R_4 والجدول R_5 فى الشكل الطبيعى «بويس - كود»، ولكن الجدول R_2 ليس كذلك. يتم اختيار إحدى الاعتماديات الوظيفية المنطبقة على R_2 وفى الوقت نفسه مخالفة للشكل الطبيعى «بويس - كود»، ولتكن:

$$PNO \rightarrow \{PName, Budget\}$$

تطبق قاعدة التجزئة $D \leftarrow (D - Q) \cup (Q - Y) \cup (X \cup Y)$ على الجدول R_2، كما يلى:

$$D \leftarrow (R_4 \cup R_5) \cup (R_6) \cup (R_7)$$

بحيث إن:

R_6 (<u>ENO</u>, <u>PNO</u>, PLocation, Duration, Responsibility)

R_7 (<u>PNO</u>, PName, Budget)

- **الدورة الرابعة:** تكون جداول التجزئة حتى الآن كما يلى $D = \{R_4, R_5, R_6, R_7\}$

الجدول R_4 والجدول R_5 والجدول R_7 فى الشكل الطبيعى «بويس - كود»، ولكن الجدول R_6 ليس كذلك. تتم تجزئة الجدول وفق الاعتمادية الوظيفية المنطبقة على R_6 وفى الوقت نفسه مخالفة للشكل الطبيعى «بويس - كود»، وهى:

$$PLocation \rightarrow PNO$$

تطبق قاعدة التجزئة $D \leftarrow (D - Q) \cup (Q - Y) \cup (X \cup Y)$ على الجدول R_6، كما يلى:

$$D \leftarrow (R_4 \cup R_5 \cup R_7) \cup (R_8) \cup (R_9)$$

بحيث إن:

R_8 (ENO, PLocation, Duration, Responsibility)

R_9 (PLocation, PNO)

ويلاحظ فى التجزئة الأخيرة أعلاه أن حقل «موقع المشروع» (PLocation) قد أصبح جزءاً من المفتاح الرئيسى للجدول الجديد (R_8) عوضاً عن حقل «رقم المشروع» (PNO) الذى تم نقله للجدول الجديد (R_9) حتى تصبح الحقول التى تمثل تقاطع الجدولين (وهى الحقل «رقم المشروع» (PLocation)) مفتاحاً رئيسياً لأحدهما كما تنص عليه قاعدة التجزئة التى تتحلى بخاصية السجلات غير الزائفة، وخاصة أنه يمكن تحديد رقم المشروع من خلال موقعه حسب الاعتمادية الوظيفية التى أدت إلى تجزئة الجدول.

- **المخرجات:** يتم التوقف لعدم وجود أى جدول مخالف للشكل الطبيعى «بويس – كود» ضمن مجموعة جداول التجزئة D. وتكون النتيجة النهائية لمجموعة الجداول كما يلى:

D = { R_4(ENO, EName, Title),
R_5(Title, Salary),
R_7(PNO, PName, Budget),
R_8(ENO, PLocation, Duration, Responsibility),
R_9(PLocation, PNO)
}

٦-١-٦ الشكل الطبيعى الرابع ((Fourth Normal Form (4NF)):

عندما يكون جدول ما فى الشكل الطبيعى «بويس – كود»، فإن مثل هذا الجدول لن يحتوى على أى من مشكلات التعديل، التى سبق أن أوضحناها أعلاه، بسبب الاعتماديات الوظيفية. إلا أن مثل هذا الجدول قد يحتوى على مشكلات أخرى نتيجة

لمـــا يعرف بـ «الاعتماديات متعددة القيـــم» ((Multivalued Dependencies (MVD)). فعلى سبيل المثال، لنفترض وجود جدول بمسمى «المواد الدراسية»، كما يلى:

COURSE

Course_ID	Instructor_Name	Textbook
Physics	Alsaleh	Physics Fundamentals
Physics	Alsaleh	Understanding Physics
Physics	Alsaleh	Introduction to Physics
Physics	Alhamad	Physics Fundamentals
Physics	Alhamad	Understanding Physics
Physics	Alhamad	Introduction to Physics
Physics	Albandar	Physics Fundamentals
Physics	Albandar	Understanding Physics
Physics	Albandar	Introduction to Physics
Chemistry	Elmasri	Introduction to Chemistry
Chemistry	Elmasri	Basic Chemistry
Mathematics	Albader	Basic Mathematics

إن الجـــدول أعلاه يتكـــون من ثلاثة حقول تمثل المادة الدراســـية وهـى: رمز المادة الدراسية (Course_ID)، واسم عضو هيئة التدريس الذى يقوم بتدريس المادة الدراسية (Instructor_Name)، والمرجـــع العلمى المســـتخدم فى تدريس المـــادة (Textbook). ولأن الحقول الثلاثة التى يتكون منها الجدول تمثل مفتاحه الرئيســـى، فإن الجدول بالشكل الطبيعـــى «بويس - كود». ومع أنه لا توجد فى الجدول أى اعتماديات وظيفية تخالف شـــروط الشكل الطبيعى «بويس - كود»، وأن الاعتمادية الوظيفية (الظاهرة) الوحيدة هى تلك المتعلقة بالمفتاح الرئيسى، كما يلى:

{Course_ID, Instructor_Name, Textbook} → {Course_ID, Instructor_Name, Textbook}

إلا أن الجدول يعانى مشكلات التعديل، كما يلى:

١- **مشـكلة الحذف:** لو تم حذف عضو هيئة التدريس الذى اسمه "Albader" فإننا ســـنفقد أيضاً مسمى المرجع الدراسى المســـتخدم فى تدريس مادة الرياضيات (Mathematics).

٢- **مشـكلة الإضافـة:** لو أردنا إضافة مرجـــع إضافى لمادة الفيزياء (Physics) فإننا

سنضطر إلى إضافة ثلاثة سجلات بحيث يضاف سجل جديد لكل عضو من أعضاء هيئة التدريس الثلاثة الذين يقومون بتدريس مادة الفيزياء.

٣- **مشكلة التحديث:** لو أردنا تحديث مسمى أحد مراجع مادة الفيزياء فإننا سنضطر لإجراء عملية التحديث فى ثلاثة سجلات. فعلى سبيل المثال، لو أردنا تحديث المرجع "Introduction to Physics" ليصبح فى طبعته الثانية "Introduction to Physics (2ed)". فإنه يجب تغيير هذا المسمى فى ثلاثة سجلات، وذلك لكل عضو من أعضاء هيئة التدريس الذين يقومون بتدريس مادة الفيزياء.

وتظهر الاعتماديات متعددة القيم عندما يكون فى الجدول ثلاثة حقول على الأقل بحيث إن أحد هذه الحقول وليكن «حقل١» يحدد مجموعة محددة من قيم حقل ثان، وليكن «حقل٢»، كما أن «حقل١» يحدد مجموعة محددة من حقل ثالث وليكنً «حقل٣» وأن مجموعة قيم «حقل٢» و«حقل٣» مستقلتان. ففى المثال أعلاه، يحدد رمز المادة الدراسية مجموعة محددة من قيم أسماء أعضاء هيئة التدريس، وهى لأولئك الذين يقومون بتدريس المادة الدراسية، كما أنه يحدد مجموعة محددة من المراجع العلمية المرتبطة بكل مادة دراسية، وأن قيم مجموعة أسماء أعضاء هيئة التدريس الذين يقومون بتدريس مادة دراسية مستقلة عن قيم المراجع العلمية الخاصة بالمادة الدراسية. وتظهر مثل هذه الاعتماديات المتعددة القيم فى الغالب عندما يتكون الجدول من مجموعة من الحقول تكون فى مجملها مفتاحاً رئيسياً للجدول.

ولتطبيع أى جدول بحيث يكون فى الشكل الطبيعى الرابع، فإنه يجب أن يكون الجدول فى الشكل الطبيعى «بويس - كود»، وفى الوقت نفسه، لا يحتوى على أى اعتماديات متعددة القيم. ويمكن النظر إلى الجدول السابق كما يلى:

COURSE

Course_ID	Instructor_Name	Textbook
Physics	Alsaleh Alhamad Albandar	Physics Fundamentals Understanding Physics Introduction to Physics
Chemistry	Elmasri	Introduction to Chemistry Basic Chemistry
Mathematics	Albader	Basic Mathematics

يوضح الجدول أعلاه كما لو كان الجدول الأصلى مكوناً من ثلاثة حقول، اثنان منها (وهما حقل اسم عضو هيئة التدريس وحقل المرجع العلمى) حقلان متعددا القيم، وأن الحقل الذى يحدد مجموعة القيم التى من الممكن أن يأخذها أى من الحقلين هو حقل رمز المادة الدراسية. كم يوضح الجدول أن مجموعة قيم كلا الحقلين متعددى القيم منفصلة عن بعضها. ولتحويل الجدول إلى الشكل الطبيعى الرابع، تتم تجزئته إلى جدولين بحيث يحتوى الجدول الأول على أسماء أعضاء هيئة التدريس الذين يقومون بتدريس المواد الدراسية بمسمى «المؤهلات التدريسية» (QUALIFICATION) فى حين يحتوى الثانى على المراجع العلمية الخاصة بالمواد الدراسية بمسمى «المواد المرجعية» (REFERENCE_MATERIAL)، كما يلى:

REFERENCE_MATERIAL

Course_ID	Textbook
Physics	Physics Fundamentals
Physics	Understanding Physics
Physics	Introduction to Physics
Chemistry	Introduction to Chemistry
Chemistry	Basic Chemistry
Mathematics	Basic Mathematics

QUALIFICATION

Course_ID	Instructor_Name
Physics	Alsaleh
Physics	Alhamad
Physics	Albandar
Chemistry	Elmasri
Mathematics	Albader

ويلاحظ فى طريقة تجزئة الجدول السابقة، التى تم فيها إنشاء جدول جديد لكل مجموعة متعددة القيم، أنها لم تقم بالتغلب على مشكلات التعديل فحسب، وإنما قد قلصت عدد السجلات حيث إن الجدولين الناتجين يحتويان على أحد عشر (١١) سجلاً، فى حين يحتوى الجدول الأصلى على اثنى عشر (١٢) سجلاً. ويعنى هذا أن تحويل أى جدول إلى الشكل الطبيعى الرابع يسهم فى تقليص حجم المساحة التخزينية المطلوبة لتخزين الجدول، وخاصة عندما تكون مجموعة القيم التى من الممكن أن تكون عليها الحقول متعددة القيم كثيرة جداً.

٦-١-١٧ الأشكال الطبيعية العليا (Higher Normal Forms):

توجد أشكال طبيعية أخرى ذات خصائص تختلف عما تم ذكره حتى الآن، مثل الشكل الطبيعى الخامس (5NF)، ولكن الحاجة الفعلية التى تتطلب تطبيع الجداول حتى تصبح بهذه الأشكال الطبيعية العليا من النادر جداً أن تظهر على أرض الواقع. لذا فإن هذه الأشكال الطبيعية تعد ذات قيمة نظرية أكثر من كونها ذات قيمة تطبيقية.

ولأن محتويات هذا الكتاب تميل إلى أن تكون ذات صبغة تطبيقية، فإننا لن نقوم باستعراض وشرح هذه الأشكال الطبيعية العليا.

٦-٢ التصميم المادى لقواعد البيانات العلاقية:

إن الهدف من التصميم المادى لنظم قواعد البيانات هو إنشاء تصميم يُمَكِّن من تخزين البيانات بشكل يوفر الأداء المناسب لنظام إدارة قاعدة البيانات على اختلاف حجم العمليات التى تنفذ على قاعدة البيانات. ويعنى هذا، وعلى خلاف التصميم المفاهيمى والتصميم المنطقى، أن التصميم المادى يوضح الكيفية التى ستخزن وتعالج فيها البيانات لا على الكيفية التى يتم من خلالها التعرف على البيانات والعلاقات فيما بينها أو طريقة تمثيلها وفق النموذج العلاقى أو نماذج البيانات الأخرى. وبناءً على هذا، يوضح هذا الجزء من الكتاب خيارات تخزين قيم الحقول وكيفية تحديد الخيار المناسب. ويوضح هذا الجزء أيضاً أن الجداول التى تم تطبيعها (أو المطبعة) قد لا تكون الشكل الأمثل الذى من المفترض أن تخزن فيه فى ملفات قاعدة البيانات، مما قد يستدعى فك التطبيع لتحسين أداء نظام قاعدة البيانات. كما يقارن هذا الجزء بين تنظيمات الملفات وعلى طرق استخدام الفهارس التى تسهم فى تسريع استرجاع البيانات من قاعدة البيانات. ويجب فى أثناء التصميم المادى لقاعدة البيانات توخى الحذر الشديد؛ إذ إن الخيارات التى يتم اتخاذها فى هذه المرحلة تؤثر فى فاعلية الوصول للبيانات، وأمن البيانات، وسهولة التعامل مع قاعدة البيانات من قبل المستفيدين.

٦-٢-١ عملية التصميم المادى:

إن الهدف الرئيسى من عملية التصميم المادى لقواعد البيانات هو التمكن من معالجة البيانات بشكل فعال، وذلك من خلال تقليص الوقت اللازم الذى تحتاج إليه التعليمات الصادرة من قبل المستفيدين (وبرامج التطبيقات) للتفاعل مع البيانات المخزنة فى قاعدة البيانات. وعلى الرغم من أن المساحة التخزينية المستخدمة لتخزين قاعدة البيانات تعد مهمة فى بعض الأحيان إلا أنها بدأت تفقد أهميتها فى السنوات الأخيرة، وذلك للتقليل المستمر من تكلفة المساحات التخزينية للحاسبات الآلية. لذلك فإن مرحلة التصميم المادى تركز عادة، وبشكل أكبر، على التسريع فى عملية التعامل مع الملفات التى تستخدمها قواعد البيانات، وعلى البيانات المخزنة فى قاعدة البيانات دون تركيز كبير على الفاعلية فى استخدام المساحة التخزينية المتاحة.

وتتطلب عملية التصميم المادى للملفات وقواعد البيانات بعض المعلومات التى تساعد على التصميم المادى الجيد، ومن المفترض أن يكون قد تم جمعها فى أثناء مراحل سابقة لتصميم النظام المعلوماتى. ومن هذه المعلومات ما يلى (Hoffer et al, 2002):

- علاقات (أو جداول) قاعدة البيانات بعد تطبيعها مع تقدير أحجام هذه العلاقات، أى تقدير عدد السجلات فى جداول قاعدة البيانات.
- تعريف لكل حقل من حقول قاعدة البيانات.
- توصيف للتوقيت والمكان اللذين يحددان متى وأين يتم التعامل مع البيانات، سواء من خلال عمليات الإدخال أو الاسترجاع أو الحذف أو التحديث مع تحديد لأعداد هذه العمليات على البيانات.
- التوقعات أو المتطلبات المتعلقة بالوقت اللازم للنظام للتعامل مع التعليمات المصدرة إليه (Response Time)، وبأمن البيانات، وبالتخزين الاحتياطى، وبالوقت اللازم للاستعادة بعد حدوث الأعطال، وبتناسق (أو تكامل) البيانات.
- توصيف للتقنيات المستخدمة فى بناء قاعدة البيانات متضمناً ذلك نظام التشغيل ونظام إدارة قاعدة البيانات.

وتحتاج عملية التصميم المادى إلى اتخاذ عدد من القرارات المهمة التى تؤثر فى تكامل وأداء نظم التطبيقات التى ستتعامل مع قاعدة البيانات. ومن أهم هذه القرارات، ما يلى:

- اختيار هيئة البيانات (Data Format) التى ستخزن عليها بيانات كل حقل من حقول قاعدة البيانات التى وردت فى التصميم المنطقى لقاعدة البيانات. ويتم اختيار هيئة البيانات بأقل ما يمكن من مساحة تخزينية وبأكثر ما يمكن من تكامل للبيانات.
- تجميع حقول البيانات الواردة فى التصميم المنطقى على هيئة سجلات مادية. وعلى الرغم من أن تجميع الحقول حسب ورودها ضمن أعمدة الجداول التى تم تصميمها فى مرحلة التصميم المنطقى لقاعدة البيانات ضمن سجلات مادية قد يبدو أمراً منطقياً إلا أنه ليس دائماً الشكل الأمثل لتجميع الحقول بشكل مباشر ضمن السجلات المادية.
- ترتيب السجلات ذات الهياكل المتشابهة فى الذاكرة الثانوية بشكل يساعد على تخزينها، وتحديثها، واسترجاعها بشكل سريع سواء كان تنفيذ هذه العمليات على سجلات منفردة أم على مجموعات منها.

- اختيار فهارس تساعد على تخزين وربط ملفات قاعدة البيانات بشكل يجعل استرجاع البيانات أكثر فاعلية.
- وضع إستراتيجيات للتعامل مع الاستفسارات التى تنفذ على قاعدة البيانات بحيث تحسن أداء تنفيذ هذه الاستفسارات وتستفيد من تنظيم ملفات قاعدة البيانات والفهارس التى تم تعريفها.

٦-٢-١-١ تصميم الحقول:

يعد الحقل أصغر وحدة تخزين للبيانات التى يمكن التعامل معها من قبل لغات البرمجة أو نظم إدارة قواعد البيانات. ومن أهم الأمور التى تجب مراعاتها فى أثناء تصميم حقول البيانات التى ستتضمنها قاعدة البيانات نوعية البيانات (Data Type)، والتحكم فى تناسق البيانات (Data Integrity) التى ستحتويها الحقول، والكيفية التى سيقوم من خلالها نظام إدارة قاعدة البيانات بالتعامل مع القيم غير المدخلة (أو المفقودة) (Missing Data).

إن اختيار نوعية البيانات المناسبة لحقول قاعدة البيانات له أربعة أهداف رئيسية تختلف أهميتها النسبية باختلاف التطبيقات التى تتعامل معها، وهى:

- تقليص المساحة التخزينية المستخدمة من قبل الحقل.
- التمكن من تمثيل جميع القيم التى من الممكن أن يحتوى عليها الحقل.
- القدرة على تحسين تكامل البيانات.
- التمكن من دعم جميع العمليات التى قد تجرى على القيم المخزنة فى الحقل.

واختيار نوعية البيانات المناسبة لحقل ما يمكِّن من تمثيل كل القيم التى من الممكن أن يأخذها الحقل الذى يقابله فى النموذج المفاهيمى وبأقل قدر ممكن من المساحة التخزينية، مع عدم إمكانية تمثيله للقيم غير المسموح بها (أى تحسين تكامل البيانات). كما أن اختيار نوعية البيانات المناسبة للحقل يمكن من دعم العمليات التى قد تجرى على القيم المخزنة فى الحقل مثل إجراء العمليات الحسابية إذا كان الحقل مُمَثلاً لبيانات عددية أو إجراء العمليات الخاصة بالحروف (String Manipulation) إذا كان الحقل ممثلاً لبيانات حرفية.

وفى العديد من نظم إدارة قواعد البيانات يمكن التحكم فى تكامل البيانات من خلال وضع قيود على الحقول ضمن هياكل الحقول نفسها، ومن ثم فرض هذه القيود

علـى الحقول من قبل نظـم إدارة قواعد البيانات. وتحدد نوعيـة البيانات التى يتم اختيارهـا لحقل ما من فرض نوع واحد من قيـود تكامل البيانات بحيث تكون بيانات الحقـل حرفيـة أو عددية (بالإضافة لطول الحقـل). إلا أن هنالك أنواعاً أخرى من القيود التى يمكن فرضها على الحقول، ومن أكثرها شيوعاً ما يلى:

- **القيمـة الافتراضيـة (Default Value)**: هى القيمة التى سـيأخذها الحقل عند عدم إدخال قيمة له فى أثناء إدخال سـجل جديـد. وتمكن القيم الافتراضية للحقول من إدخال البيانات بشـكل سـريع من قبل المسـتفيدين، وذلك عندما تكون الغالبية العظمى من السـجلات التى يتم إدخالها لها قيمة تساوى القيمة الافتراضية للحقل، ومن ثم يمكن تخطى عملية إدخال قيم لحقول السـجلات التى تساوى القيمة الافتراضية للحقل، مما يسهم فى الإسراع فى عملية إدخال البيانات.
- **التحكـم فـى مدى القيـم (Domain Values)**: يمكن فـرض قيود على مجموعة القيم التى من الممكن أن تأخذها قيم حقل ما. ويمكن أن يكون المدى عددياً يتم تحديده بقيمة دنيا وقيمة عليا أو مجموعة من القيم المحددة سواء كانت عددية أو سلاسل حرفية أو التواريخ.
- **التحكم فى القيم غير المعرفة (Null Values)**: قد يُسمح لبعض الحقول أن تأخذ القيمـة غير المعرفة، فى حين لا يسـمح لحقول أخـرى أن تأخذ هذه القيمة. والتحكـم فى القيم غير المعرفة للحقول يسـاعد على تكامـل البيانات. فعلى سبيل المثال، قد تمنع السياسة العامة لإحدى الجامعات إضافة أية مادة دراسية جديدة ما لم يكن للمادة الدراسـية اسـم (مثل مقدمة فى الحاسـب الآلى أو نظم قواعد البيانات). كذلك هو الحال بالنسبة لسجلات الموظفين، فقد تمنع السياسـة العامة لمنظمة ما من إضافة أى سجل لموظف جديد ما لم يتم إدخال قيمة فى حقل رقم الموظف.
- **السـلامة المرجعية (Referential Integrity)**: تعد السلامة المرجعية أحد أشكال التحكم فى المدى، إذ إن القيم التى بإمكان الحقل أن يأخذها يجب أن تكون من ضمن قيم حقل آخر موجود فى سجل آخر ضمن الجدول نفسه أو جدول آخر. ويعنى هذا أن مدى القيم المفروض على حقل من هذا النوع ذو طبيعة ديناميكية تختلف مع مرور الزمن باختلاف القيم التى يحتويها الحقل الآخر الذى تستمد منه هذه القيم عوضاً عن كون مدى قيم الحقل ثابتة ومعرفة بشكل مسبق.

٦-٢-١-٢ تصميم السجلات وعملية فك التطبيع:

أثناء عملية التصميم المنطقى يتم تجميع الحقول ضمن صفوف فى جداول قاعدة البيانات بحيث يمثل كل صف حالة معينة يتم تحديدها من خلال المفتاح الرئيسى للجدول الذى يحتوى على الصف. وعلى النقيض من ذلك فإن التصميم المادى للحقول يعنى وضعها بشكل يجاور بعضها بعضاً فى الذاكرة الثانوية بحيث يمكن استرجاعها وكتابتها كوحدات (Pages) من قبل نظام إدارة قاعدة البيانات. لذلك فإن التصميم المادى للحقول يتطلب اختيار ترتيب الحقول بشكل متجاور، حتى يمكن (١) الاستفادة من الذاكرة الثانوية بشكل فعال، وحتى يمكن (٢) معالجة البيانات بشكل سريع.

وتتأثر درجة الاستفادة من الذاكرة الثانوية بشكل كبير بمقاس الحقول وهيكلية الذاكرة الثانوية، فنظم التشغيل تتعامل مع البيانات المخزنة فى الذاكرة الثانوية (سواء بقراءتها أو كتابتها) كوحدات تسمى صفحات (Pages). والصفحة تمثل كمية البيانات التى بإمكان نظام التشغيل قراءتها أو كتابتها من خلال عملية إدخال (للذاكرة الرئيسية) أو إخراج (من الذاكرة الرئيسية) واحدة. ومقاس الصفحة مقاس ثابت يتم تحديده من قبل مديرى قواعد البيانات بحيث تتم الاستفادة القصوى من حجم الذاكرة الرئيسية لجهاز الحاسب من قبل جميع التطبيقات التى تنفذ عليه. وبحسب طبيعة الحاسب الآلى المستخدم فإنه قد يسمح بتوزيع محتويات السجل الواحد على صفحتين أو قد لا يسمح بذلك. ويعنى هذا أنه قد يتم إهدار الكثير من المساحة التخزينية فى حال كان مقاس الصفحة الواحدة لا يمثل حاصل ضرب طول السجل الواحد بعدد صحيح؛ مما يترك بعض المساحة التخزينية المهدرة فى نهاية كل صفحة. ويسمى عدد السجلات التى بالإمكان تخزينها فى الصفحة الواحدة بمعامل الكتلة (Block Factor (BF)). وعندما تكون المساحة التخزينية المتوافرة فى الحاسب الآلى قليلة (نسبياً) وفى الوقت نفسه لا يمكن أن توزع محتويات السجلات على أكثر من صفحة فإن إنشاء أكثر من ملف لجدول ما وتوزيع حقول سجلات الجدول عليها يقلص من المساحة التخزينية المهدرة.

على الرغم من أن تقليص المساحة التخزينية المهدرة يعد أحد العوامل المهمة فى عملية التصميم المادى لقاعدة البيانات، إلا أن عامل التسريع فى معالجة البيانات يفوق فى أهميته العامل الآخر. وتعتمد سرعة معالجة البيانات على مدى قرب البيانات المرتبطة بعضها ببعض (والتى تتعامل معها التعليمات الواردة للنظام بشكل متكرر) فى وسائل التخزين الثانوية. ففى العادة لا يتم استخدام حقول الجداول كافة للإجابة

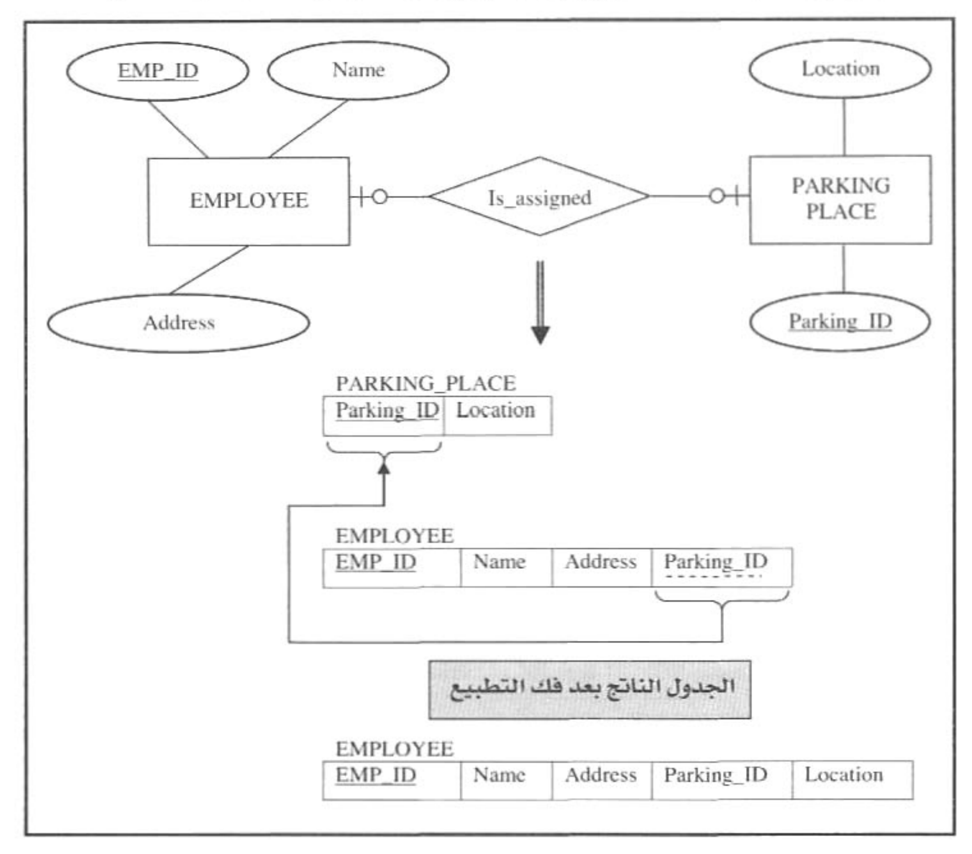
عن العمليات التى تتفاعل مع قاعدة البيانات، وإنما يتم استخدام حقول بيانات تابعة لأكثر من جدول عند تنفيذ العمليات التى ترد لنظام إدارة قاعدة البيانات. وبناءً على ذلك فإن الجداول التى تم تطبيعها فى أثناء عملية التصميم المنطقى لقاعدة البيانات التـى تقلل من تكرارية البيانات، ومن ثم الحد من أخطاء التعديل قد لا تكون الشكل الأمثل الذى يؤدى إلى معالجة فعالة للبيانات إذا ما تم بناؤها مادياً بالشكل نفسه الذى صممت عليه منطقياً. وقد أوضحت بعض الدراسات أن الفاعلية فى معالجة بيانات قاعدة بيانات مبنية مادياً حسب بنائها المنطقى المطبع بشكل كامل قد يكون مكلفاً وبشكل كبير مقارنةً ببناء قاعدة البيانات نفسها، ولكن من خلال تطبيعها بشكل جزئى (Inmon, 1988). وعلى الرغم من أن نتائج مثل هذه الدراسات تعتمد على محتويات قاعدة البيانات وطبيعة العمليات التى تنفذ عليها، إلا أن مثل هذه الدراسات توضح ضرورة توخى الحرص عند بناء قاعدة البيانات مادياً من حيث بناؤها بشكل يتوافق مع تطبيعها بشكل كامل أو بناؤها بشكل أقل درجة فى التطبيع مقابل تخفيض تكلفة معالجة التعليمات التى ترد لنظام إدارة قاعدة البيانات.

وتهدف عملية فك التطبيع (Denormalization) إلى تحويل العلاقات (أو الجداول) المطبعة إلى سجلات يتم تخزينها مادياً بشكل أقل تطبيعاً. وقد ينتج عن عملية فك التطبيع تجزئة حقول كل سجل من سجلات علاقة ما إلى مجموعة من السجلات عوضاً عن تخزينها مادياً كسجل واحد، أو أن يتم دمج حقول سجلات تابعة لأكثر من علاقة فى سجل مادى واحد أو كلا الاثنين معاً. وعلى الرغم من وجود الكثير من السلبيات التى قد تنتج عن عملية فك التطبيع إلا أن هذه السلبيات تتلاشى إذا ما تمت العملية بشكل دقيق ونتج عنها زيادة كبيرة فى سرعة معالجة البيانات.

ويوجد هناك عدد من الحالات المعروفة التى قد تتطلب إجراء عمليات فك التطبيع، وفيما يلى ثلاث من هذه الحالات (Rogers, 1989):

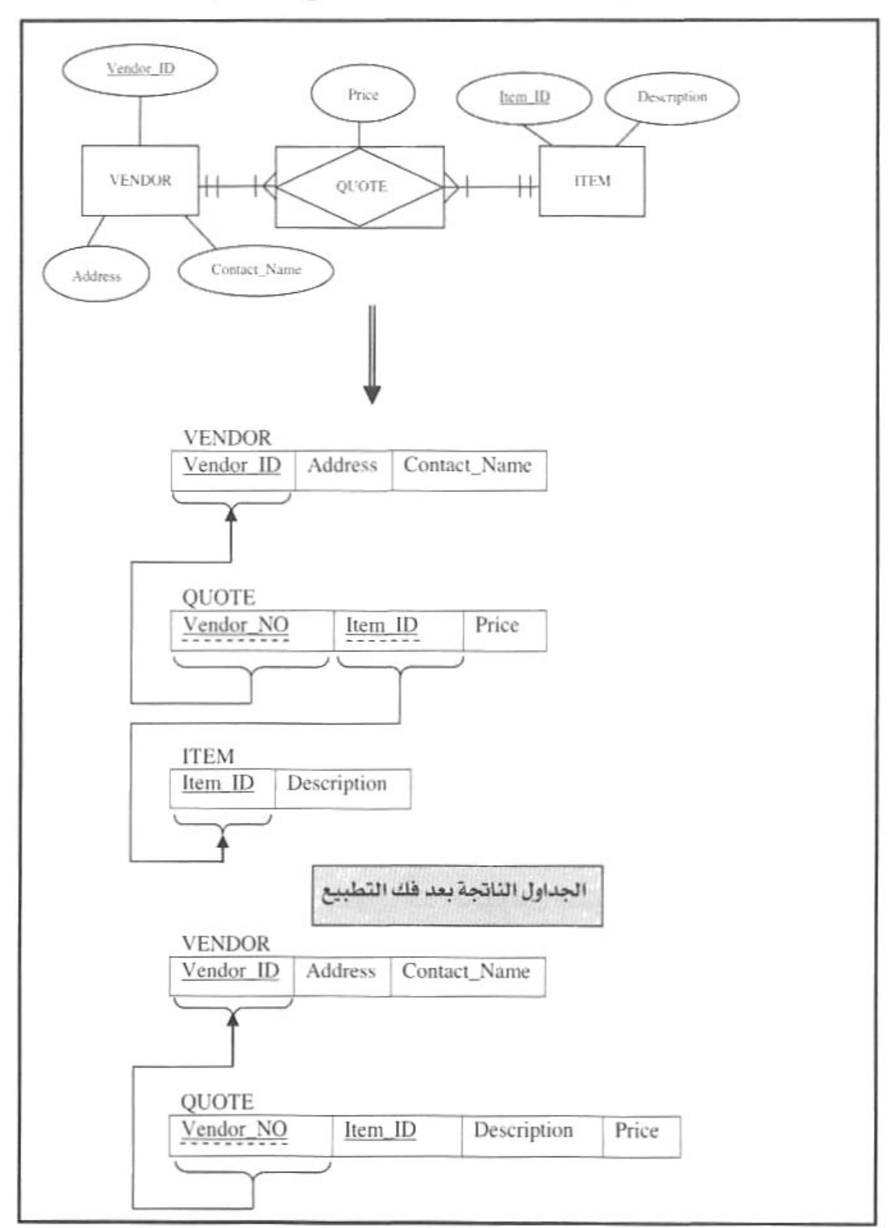
١- **جدولان تربط بينهما علاقة واحد - واحد:** يُفضَّل فك التطبيع فى هذه الحالة حتى لو كان أحد الجدولين ذا علاقة اختيارية، وذلك عندما يكون هناك ارتباط بين سجلات الجدولين فى غالبية الأحيان. ويتم فى هذه الحالة إنشاء سجل واحد ضمن جدول واحد عوضاً عن سجلين فى جدولين مختلفين. وبهذه الطريقة يتم الاستغناء عن عمليات الربط التى يجب إجراؤها للحصول على البيانات المخزنة فى الجدولين للحصول على بيانات السجلات التى تربط بينهما العلاقة. ويوضح الشكل رقم (٦-٨) مثالاً لمثل هذه الحالة؛ إذ يوجد لكل موظف موقف سيارات واحد على الأكثر، وأن كل موقف سيارات مخصص لموظف واحد على الأكثر.

شكل رقم (٦-٨): فك التطبيع بين جدولين تربط بينهما علاقة واحد - واحد



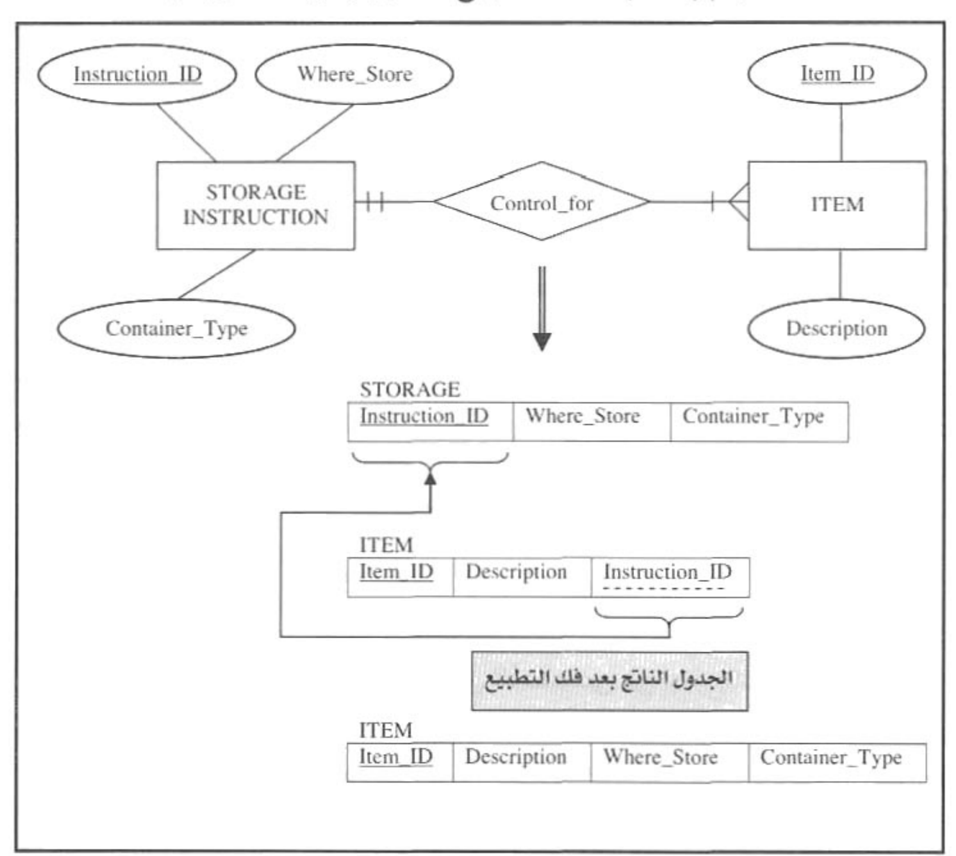
٢- **علاقة متعدد - متعدد (أى علاقة مشاركة) بين جدولين ولا يوجد للعلاقة مفتاح خاص بها:** قد يكون من المفيد دمج حقول أحد الجدولين اللذين تربط بينهما العلاقة ليكونا من ضمن الجدول الذى يمثل العلاقة. وبهذه الطريقة يمكن إجراء عملية ربط واحدة بين الجدولين الناتجين عوضاً عن إجراء عمليتى ربط بين ثلاثة جداول. وتتجلى أهمية هذه الطريقة عندما تكون غالبية العمليات المنفذة على الجداول الثلاثة تتطلب عمليات ربط بينها. ويوضح الشكل رقم (٦-٩) مثالاً لهذه الحالة.

شكل رقم (٦-٩): فك التطبيع عند وجود علاقة متعدد - متعدد (أى كينونة مشاركة) بين جدولين ولا يوجد للعلاقة مفتاح خاص بها



٣- **بيانات مرجعية:** توجد البيانات المرجعية عندما تكون هناك كينونة فى الجانب ذى التعدديـــة الأحادية من علاقة ذات تعددية واحد - متعدد، وهذه الكينونة لا ترتبط بأية علاقات أخرى مع الكينونات الأخرى المكونة لقاعدة البيانات. فى هذه الحالة يفضـــل دمج الجدولين الممثلين للكينونتين ضمن جدول واحد، وخاصة عندما يكون عدد الحالات فى الجانب المتعدد المرتبطة بالجانب الأحادى قليلة نسبياً.

شكل رقم (٦-١٠): فك التطبيع عند وجود بيانات مرجعية



يوضح الشـــكل رقم (٦-١٠) مثالاً للحالة المذكورة أعلاه، إذ إن كل «عنصر» (Item) لـــه طريقة فى التخزين (STORAGE_INSTRUCTION) تتمثل فى مكان تخزين العنصر (Where_Store) ونـــوع الإنـــاء (Container_Type). وعندما يكون عـــدد حالات العناصر

قليلاً نسـبياً، وتكون «طريقة التخزين» (STORAGE_INSTRUCTION) مرتبطة بكينونة «العناصـر» (ITEM) فقط، فإنه يفضل دمج جدول العناصر مع جدول طريقة التخزين ضمن جدول واحد وهو جدول العناصر (ITEM).

إن المواقع السـابقة التى قد تسـتدعى عمليات فك التطبيع تهدف إلى الاسـتغناء عـن عمليات ربط بين جداول قاعدة البيانات التى تسـتنزف الوقت الكثير لتنفيذها، وخاصـة عند احتواء الجداول على أعداد كبيرة من السـجلات. ويتم ذلك من خلال دمـج الجداول بعضها مع بعض. وعلى النقيـض من ذلك فإنه يمكن فك التطبيع من خلال تقسـيم جدول ما إلى أكثر من جـدول عوضاً عن دمجه مع جدول آخر كما فى الحالات السـابقة. وهناك ثلاثة أنواع من تقسـيم الجداول وهى: التقسيم الأفقى، والتقسـيم الرأسى، والتقسـيم الذى يمزج بين الاثنين (أفقياً ورأسـياً معاً). ويهدف التقسـيم الأفقى لجدول ما إلى تجميع السـجلات التى تشترك فى قيمة خاصية (أو حقل) معين ضمن جدول واحد، بحيث تتم غالبية الاستفسارات التى ترد للنظام وفقاً لقيمة هذه الخاصية. فعلى سـبيل المثال، قد يتم تقسيم جدول الموظفين العاملين فى منظمـة ما إلى أكثر من جدول وفقـاً للإدارات التى يعمل فيهـا الموظفون. فى هذه الحالة يكون من الأسرع الاستجابة للتعليمات التى ترد للنظام والتى تتعامل مع سجلات الموظفـين العاملين فى كل إدارة على حدة. كذلك هو الحال بالنسـبة لجدول العملاء المتعاملين مع شركة ما؛ إذ يمكن تقسيم ملف العملاء حسب المناطق التى يقطنها هؤلاء العملاء. وفى مثل هذه الحال يكون من الأسرع الاستجابة للاستفسارات التى تتعامل مع العملاء وفق المناطق التى يقطنونها.

أما التقسـيم الرأسـى فيقوم بتوزيع الحقول التى يحتويها جدول ما على أكثر من جدول مع تكرار المفتاح الرئيسـى للجدول الأساسـى فى جميع الجداول الناتجة من عملية التقسـيم. ومن أمثلة التقسـيم الرأسى تقسـيم جدول الموظفين إلى جدولين: جـدول يحتوى على المفتاح الرئيسـى لجدول الموظفين، بالإضافـة إلى الحقول التى تحتوى على البيانات المتعلقة بالجوانب الإدارية للموظفين، فى حين أن الجدول الثانى يحتـوى على المفتاح الرئيسـى لجدول الموظفين، بالإضافـة إلى الحقول التى تحتوى علـى البيانات المتعلقة بالجوانب المالية للموظفين. وبهذه الطريقة يمكن الاسـتجابة للتعليمات المتعلقة بكل نوع من البيانات بشـكل أسرع من الرد عليها عند تخزين جميع البيانات المتعلقة بالموظفين ضمن نفس الجدول.

ومن الميزات التى يتحلى بها كلا التقسيمين أعلاه إعطاء صلاحيات للمستفيدين على الجداول الناتجة بعد عملية التقسيم، وذلك عوضاً عن إعطائهم صلاحيات على الجداول بشكلها الكامل (قبل عملية التقسيم)؛ مما يسهم فى دقة إعطاء الصلاحيات، ومن ثم تحسين درجة أمن وسرية المعلومات.

أما النوع الثالث للتقسيم فيتضمن التقسيم الأفقى والتقسيم الرأسى. فعلى سبيل المثال، من الممكن تقسيم جدول الموظفين أفقياً حسب الإدارات التى يعمل فيها الموظفون ورأسياً حسب بياناتهم الإدارية والمالية. وتتجلى فائدة هذا النوع من التقسيم بشكل خاص فى قواعد البيانات الموزعة التى سنتطرق إليها فى الفصل التاسع.

٦-٢-١-٣ تنظيم الملفات (File Organization):

يستخدم تنظيم الملفات لترتيب السجلات التى ستخزن فى ملفات الذاكرة الثانوية. ويتم اختيار أحد تنظيمات الملفات فى نظم إدارة قواعد البيانات لملف ما آخذين بعين الاعتبار سبعة عوامل مهمة، وهى (Hoffer et al, 2002):

- السرعة فى استرجاع البيانات.
- السرعة فى الحصول على نتائج معالجة التعليمات.
- الفعالية فى استخدام الذاكرة.
- الحماية من الأعطال أو فقد البيانات.
- التقليل من الحاجة إلى إعادة تنظيم السجلات فى الملف.
- التمكن من التحكم فى التوسع فى حجم البيانات.
- تأمين البيانات من الاستخدامات غير المسموح (أو المصرح) بها.

وعادة ما تتعارض العوامل السابقة بعضها مع بعض عند اختيار تنظيم معين للملفات ويجب اختيار التنظيم الذى يوفر التوازن المناسب بين المعايير السابقة للاستفادة القصوى من المصادر المتاحة فى النظام. وفيما يلى شرح (مقتضب) لأهم تنظيمين للملفات وهما: الملفات المتسلسلة، والملفات المفهرسة.

الملفات المتسلسلة: يتم فى الملفات التى تنظم بشكل متسلسل (Sequential Files) تخزين السجلات بشكل متسلسل، الواحد تلو الآخر، حسب قيمة مفتاح أولى (Primary Key). وللوصول إلى سجل ما يتم المرور على السجلات المخزنة فى الملف من البداية،

وبشــكل متسلسل، حتى الوصول إلى الســجل ذى قيمة المفتاح المطلوب. ومن الأمثلة الشــهيرة للملفات المتسلســلة دليل الهاتف الذى يتم فيه ترتيب أسماء المشتركين فى خدمة الهاتف أبجدياً (وبشكل تصاعدى) حسب أسماء عائلاتهم. وعندما يتم البحث عــن رقم هاتف أحد المشــتركين، فإنه لا بد أن يتم المرور على جميع الســجلات التى تســبق اســم المشــترك قيد البحث عن رقم هاتفه حتى الوصول إلى الاسم المطلوب. وتجدر الإشــارة إلا أن هــذا النوع من الملفات لا يتم اســتخدامه من قبل نظم قواعد البيانات، وذلك لعدم مرونته فى الوصول إلى الســجلات بشــكل فعال. فعلى ســبيل المثال، تحتاج نظم إدارة قواعد البيانات إلى فحص (n/2)، فى المتوســط، من ســجلات جدول ما للوصول إلى الســجل ذى المفتاح (k) عندما يكون عدد السجلات المخزنة فى الملف هى لجدول عدد ســجلاته (n)؛ وذلك لأن السجلات التى يجرى البحث عنها قد تكون فى بدايات الملف أحياناً، وقد تكون فى نهايات الملف أحياناً أخرى بحســب قيم مفاتيحها. ولكون المتوســط العام للوصول إلى الســجلات المطلوبة باستخدام الملفات المتسلسلة يعد عالياً نسبياً مقارنة بأنواع الملفات الأخرى، فإن هذا النوع من الملفات قد يتم اســتخدامه لأغراض النسخ الاحتياطى، ولكنه لا يستخدم باعتباره طريقة لتخزين البيانات فى قواعد البيانات.

الملفات المفهرســة: يتم تنظيم الســجلات فى الملفات المفهرسة إما بشكل متسلسل منظم حسب قيمة مفتاح أولى ما أو دون تسلسل معين للسجلات فى الملف. ويتم بناء فهرس للملف يسمح بالوصول إلى السجلات المطلوبة فى الملف. والفهرس (فى أبسط صوره) هو جدول يحتوى على عمودين: أحدهما يحتوى على قيم مفاتيح السجلات فى الملف، فى حين يحتوى العمود الثانى على مؤشرات (Pointers) تبين مواقع (أو عناوين) الســجلات فى الملف. ويمكن تشبيه الملفات المفهرسة بفهارس المطبوعات العلمية فى المكتبات: إذ إن كل فهرس من فهارس المكتبات يحتوى على مفتاح مثل اســم المؤلف أو دار النشــر أو الموضوع، ويحتوى على قيمــة توضح موقع المطبوعة ضمن ثنايا المكتبة. وبذلك فإن كل ســجل فى الفهــرس يحتوى على قيمة لمفتاح ومؤشــر يدل على موقع السجل ذى نفس قيمة المفتاح فى الملف. وعندما يكون المفتاح مفتاحاً أولياً (أو رئيسياً) فإن لكل ســجل قيمة مفتاح تختلف عن بقية الســجلات فى الملف ولا يمكن أن تتكرر. أما إذا كان من الممكن أن تتكرر قيمة المفتاح فإن المفتاح يسمى ثانوياً (Secondary Key). فعلى سبيل المثال يعد مفتاح الفهرس المبنى على أرقام الطلبة مفتاحاً أولياً وذلك لعدم إمكانية تكــرر أرقام الطلبة، فى حين يعد مفتاح الفهرس المبنى على أســماء عائلات الطلبة مفتاحاً ثانوياً؛ لأنه من الممكن أن تتكرر أسماء عائلات الطلبة.

وتكمن أهمية الفهارس فى نظم إدارة قواعد البيانات، بشكل عام، فى كونها تقلص من حجم البيانات الواجب البحث فيها حتى يتم الوصول إلى السجلات المطلوبة؛ إذ إن حجـــم أى فهرس يكون عادة أقل بكثير من حجم الملف الذى بنى عليه الفهرس، وذلك لكون سجلات الملف عادة ما تكون أطول بكثير من سجلات الفهرس. ويعنى هذا أنه، وفى غالبية الأحيان، يمكن أن يوضع الفهرس بالكامل فى الذاكرة الرئيســية للحاسب الآلى والبحث عن مواقع السجلات من خلال البحث فى الفهرس. أما فى حالة عدم وجـــود فهرس فإنه يجب الرجوع للذاكرة الثانويـــة للجهاز، ولمرات عديدة، لنقل أجزاء من الملف والبحث فيها حتى يتم الوصول إلى الســجلات قيد البحث. والســبب وراء ذلــك يعود إلى كبر حجم الملفات التى تحتوى على ســجلات البيانات التى يتعذر معها وضع مثل هذه الملفات فى الذاكرة الرئيسية للحاسب الآلى دفعة واحدة. ولأن الذاكرة الثانويــة تكون عادة مبنية على تقنيات تدخل فيهــا الحركة الميكانيكية، مثل الأقراص الصلبة التى تعد الأكثر شــيوعاً فى تخزين البيانــات، فإن عملية الرجوع إلى الذاكرة الثانوية بشكل متكرر تعد مكلفة جداً فى عمليات البحث وتؤدى إلى الإطالة فى الوقت اللازم للوصول إلى السجلات المطلوبة مما يؤثر فى فاعلية نظام إدارة قاعدة البيانات بشكل عام.

وكما هو فى فهارس المكتبات، على ســبيل المثال، يمكن بناء أكثر من فهرس للملف نفسه. كما تتوافر أنواع عديدة من الفهارس التى بالإمكان بناؤها على ملفات البيانات، ولكننا لن ندخل فى تفاصيل الأنواع المختلفة للفهارس هنا.

٦-٢-١-٤ إنشاء واستخدام الفهارس:

تتطلب غالبية التعليمات التى تنفذها نظم إدارة قواعد البيانات الوصول إلى سجل أو أكثــر يتحقق فيها شــرط معين. ومــن أمثلة هذه التعليمات معرفــة أعضاء هيئة التدريس الذين يعملون فى قســم الحاسب الآلى، أو معرفة الطلبة الذين يدرسون فى تخصــص معين، أو معرفة الطلبة ذوى المعــدلات التراكمية التى تكافئ «جيد جداً» أو أكثر. وللحصول على نتيجة مثل هذه التعليمات يجب المرور على جميع سجلات الملف الذى يحتوى على الســجلات المطلوبة، الواحد تلو الآخر، ومقارنة الحقول التى ترتبط بشرط البحث مع القيمة المفروضة على شرط البحث. وتعد عملية المرور على جميع سجلات الملف بهدف الوصول إلى النتيجة المطلوبة عملية مكلفة جداً، وخاصة عندما يكون عدد الســجلات فى الملف قيد البحث كبير جداً. لذلك يتم اســتخدام الفهارس

التى تم شــرحها (بشكل مقتضب) أعلاه للتســريع فى تنفيذ مثل هذه التعمليات التى تمثل غالبية التعليمات التى تنفذ على قاعدة البيانات.

٦-٢-١-٤-١ إنشاء الفهارس ذات المفاتيح الفريدة (Unique Key Index):

يمكن إنشاء الفهارس ذات المفاتيح الفريدة على أى حقل فى جدول بحيث إن قيمة الحقل الذى سينشأ عليه الفهرس لا يمكن أن تتكرر فى سجلات الجدول. فعلى سبيل المثال، يمكن إنشاء فهرس فريد على حقل رمز المادة الدراسية (Course_ID) فى جدول المواد الدراســية (COURSE_T) فى قاعدة بيانات الجامعــة الأهلية الذى يمثل المفتاح الرئيسى للجدول، كما يلى:

```
CREATE UNIQUE INDEX COURSE_ID_IDX ON COURSE_T (COURSE_ID);
```

وتمثل العبارة (COURSE_ID_IDX) اسم الفهرس الفريد الذى سيتم إنشاؤه لحفظ مكونات الفهرس، فى حين يمثل ما بعد العبارة (ON) اســم الجدول الذى ســيتم بناء الفهــرس عليه والحقل الذى ســيكون مفتاح الفهرس. وعندما تنفذ تعليمة الإنشــاء السابقة ســتتم فهرسة جميع سجلات جدول المواد الدراسية ضمن الفهرس. أما إذا وجد أكثر من سجل لها قيمة رقم المادة الدراسية نفسها (أى القيمة نفسها فى الحقل الذى يمثل مفتاح الفهرس) فستفشل عملية إنشاء الفهرس. وعند إنشاء الفهرس، فى حال عدم وجود ســجلات تتكرر فيها قيمة مفتاح الفهرس، فإنه سيتم أيضاً رفض أى عملية إضافة للجدول ينتج عنها سجلات تتكرر فيها قيمة مفتاح الفهرس.

وعندما يراد إنشــاء فهــرس فريد ذى مفتاح مركب من أكثــر من حقل فإنه يمكن ذلك من خلال إدراج جميع الحقول المكونة للمفتاح بعد عبارة (ON) التى تتضمن اســم الجدول. فعلى ســبيل المثال، يمكن إنشاء فهرس فريد على حقل رمز المادة الدراسية (Course_ID) ورمز المادة الدراسية المتطلبة (Prerequisite_ID) فى جدول المواد الدراسية المتطلبة (PREREQUISITE_T) اللذين يمثلان المفتاح الرئيسى لجدول، كما يلى:

```
CREATE UNIQUE INDEX COURSE_PREREQUISITE_IDX
ON PREREQUISITE_T (COURSE_ID, PREREQUISITE_ID);
```

وتقـــوم غالبية نظم إدارة قواعد البيانات العلاقية بإنشـــاء فهرس فريد، وبشـــكل تلقائى، لكل جدول يتم إنشاؤه بحيث يكون مفتاح الفهرس هو المفتاح الرئيسى للجدول. ويعنى هذا عدم الحاجة إلى تنفيذ تعليمة إنشـــاء فهرس إذا كان الحقل الذى سينشـــأ عليه الفهرس هو المفتاح الرئيسى للجدول مثل الفهارس التى تم إنشاؤها أعلاه.

٦-٢-١-٤-٢ إنشاء الفهارس ذات المفاتيح الثانوية (أو غير الفريدة) (Secondary Key Index):

تستخدم الفهارس ذات المفاتيح الثانوية عندما يراد التعامل مع حقول غير الحقول الممثلة للمفاتيح الرئيســـية للجداول. فعلى ســـبيل المثال، قد يتعامل المســـتفيدون مع جـــدول الطلبة (STUDENT_T) فى قاعدة بيانـــات الجامعة الأهلية من خلال تعليمات تحاول التعرف على أســـماء الطلبة فى تخصصات معينة، أو قد يتم التعامل مع جدول المتطلبات الدراسية من خلال تعليمات تحاول التعرف على المتطلبات الدراسية للمواد الدراســـية المختلفة. وللتســـريع من عملية تنفيذ مثل هذه التعليمات فإنه من الممكن إنشـــاء فهرس على كل حقل من الحقول المضمنة فى شـــرط الاسترجاع. فعلى سبيل المثال، يمكن إنشاء فهرس (غير فريد) على حقل تخصصات الطلبة باستخدام التعليمة التالية:

```
CREATE INDEX MAJOR_IDX ON STUDENT_T(MAJOR);
```

٦-٢-١-٤-٣ استخدامات الفهارس:

يجب فى مرحلة التصميم المـــادى لقاعدة البيانات العناية فى اختيار الحقول التى ســـيتم استخدامها فى إنشـــاء الفهارس، وذلك لوجود عملية مقايضة (trade-off) بين تحسين أداء نظام إدارة قاعدة البيانات فى تنفيذ تعليمات الاسترجاع وبين خفض أداء النظام فى تنفيـــذ عمليات الإضافة والحذف والتحديث؛ وذلك لكون العمليات الثلاث الأخيـــرة قد تتطلب تعديل محتويات الفهـــارس المبنية على الجداول التى تتعامل معها هـــذه العمليات. ومثل عمليـــات التعديل هذه فى محتويات الفهارس تضيف المزيد من التكلفـــة عند تنفيذ عمليات التعديل على محتويات قاعـــدة البيانات. لذلك فإنه يتم عادةً الاســـتخدام المكثف للفهارس عندما تتسم الغالبية العظمى من العمليات المنفذة على قاعدة البيانات فى كونها عمليات اســـترجاع. أما إذا كانت الغالبية العظمى من

العمليات التى تنفذ على قاعدة البيانات تتسم فى كونها عمليات تعديل فإنه يجب أخذ الحيطة والحكمة عند إنشـاء الفهارس. وفيما يلى بعض القواعد العامة التى تساعد على اختيار وإنشاء الفهارس:

- تعد الفهارس مفيدة للجداول الكبيرة (ذات السجلات الكثيرة فى عددها).

- تعد الفهارس مفيدة للحقول التى تظهر فى عبارة شرط الاسترجاع (Where) من تعليمة الاختيار (Select) أو فى عبارات الربط.

- تعد الفهارس مفيدة للحقـول التى تظهر فى عبارة الترتيب (Order By) وعبارة التصنيـف (Group By)، مـع ضرورة التأكد من أن نظـام إدارة قاعدة البيانات سيستخدم فعلياً الفهارس المنشأة لهذا الغرض عند تنفيذه لتعليمات الاسترجاع التـى تحتوى على عبارات الترتيب وعبـارات التصنيف: إذ إن بعض نظم إدارة قواعد البيانات قد لا تستخدم الفهارس فى مثل هذه التعليمات.

- تعد الفهارس مفيدة للحقول التى يزيد عدد القيم التى من الممكن أن تخزن فيها عن ثلاثين (٣٠) قيمة وأن عدد السـجلات المسترجعة لا تزيد على عشرين فى المائة (٢٠٪) من عدد السجلات الكلية المخزنة فى الملف.

- يجـب التأكد مـن الحد الأقصى لعدد الفهارس التى من الممكن أن تنشـأ لكل جـدول: إذ إن غالبية نظم إدارة قواعد البيانات لا تسـمح بأن يزيد هذا العدد على سـتة عشـر (١٦) فهرسـاً. كما يجب معرفة الحد الأقصى لطول مفتاح الفهـرس عند اسـتخدام المفاتيح المركبة للفهارس. فعلى سـبيل المثال، قد لا يسمح نظام إدارة قاعدة البيانات أن يزيد طول مفتاح الفهرس على مائة (١٠٠) بايت.

- يجب الحذر عند فهرسـة حقول توجد فيها القيمـة غير المعرفة (Null): إذ إن مثل هـذه القيمة لن تكون من ضمن قيم مفاتيح الفهرس فى غالبية نظم إدارة قواعد البيانات مما يسـتوجب المرور على سـجلات الملف كافة على الرغم من وجود فهرس للجدول.

وتعـد عملية اختيار الفهارس واحدة من أهم العمليات فى مرحلة التصميم المادى، إلا أنها ليسـت العملية الوحيدة، فمن العمليات الأخرى التى من شـأنها تحسين أداء

نظام إدارة قاعدة البيانات عملية تقليص المساحة التخزينية المهدرة فى الملفات، ومن ثم تقليص حجم الملفات الذى يترتب عليه تقليص عدد المرات التى يجب فيها الرجوع إلى الذاكرة الثانوية لنقل كل ملف إلى الذاكرة الرئيسية عند الحاجة إلى ذلك؛ وعملية تقليص تكلفة تنفيذ الاستفسارات (Query Optimization) التى يستخدم فيها خوارزميات مخصصة لهذا الغرض، إلا أننا لن نتطرق إلى هذين الموضوعين فى هذا الكتاب.

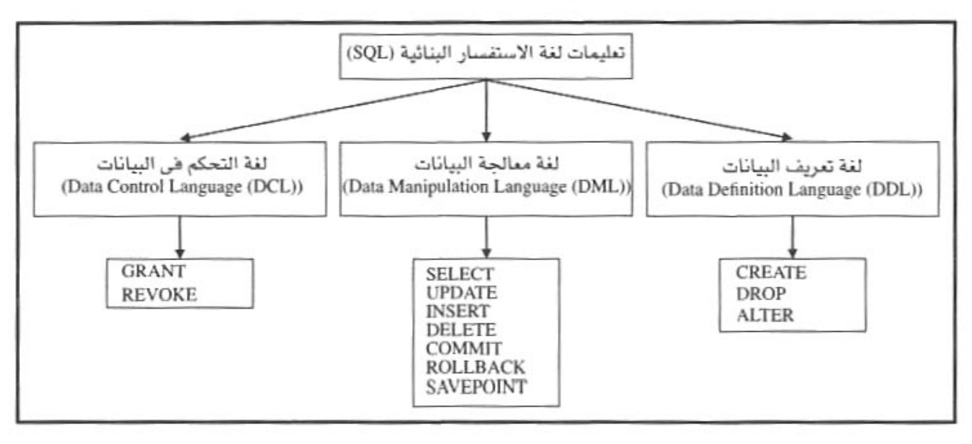
الفصل السابع

لغة الاستفسار البنائية - الجزء الأول

تعد لغة الاستفسار البنائية (Structured Query Language (SQL)) واحدة من أكثر لغات قواعد البيانات العلاقية انتشاراً. وقد تم تبنى هذه اللغة من قبل معهد المقاييس الوطنى الأمريكى (American National Standards Institute (ANSI)) ومنظمة المقاييس الدولية (International Standards Organization (ISO)). وقد تم نشر أول مقياس لهذه اللغة من قبل معهد المقاييس الوطنى الأمريكى عام ١٩٨٦م (Cannan and Otten, 1993). وتم تحديث مقياس لغة الاستفسار البنائية عام ١٩٨٩م وعام ١٩٩٢م وعام ١٩٩٩م، على التوالى. ويتكون المقياس حالياً من ثلاثة مستويات، وهى: الأساسى (Entry)، والمتوسط (Intermediate)، والكامل (Full).

وعلى الرغم من إطلاق كلمة «الاستفسار» على لغة التعامل مع قواعد البيانات العلاقية، إلا أن هذه الكلمة مجازية بمعنى أن مكونات لغة الاستفسار البنائية لا تنحصر فى التعليمات التى تقوم بعمليات الاستفسار، بل تتعدى ذلك لتحتوى على ثلاث فئات من أنواع التعليمات. تُعنى الفئة الأولى بعمليات إنشاء مكونات (أو هياكل) قاعدة البيانات، وإزالتها، والتعديل عليها. أما الفئة الثانية من أنواع التعليمات فتُعنى بالعمليات التى تقوم بالتعامل الفعلى مع البيانات المخزنة فى قاعدة البيانات، مثل عمليات الإضافة إليها، والحذف منها، والتحديث عليها. الفئة الثالثة من أنواع التعليمات تُعنى بعمليات التحكم فى تداول البيانات من قبل المستفيدين من خلال تزويدهم بالصلاحيات المناسبة لتداولها أو سحب الصلاحيات منهم. وعلى الرغم من أن كل الفئات الثلاث من التعليمات تتبع للغة واحدة وهى لغة الاستفسار البنائية، إلا أن هذه الفئات تسمى مجازاً لغات فرعية (Sub-languages) للغة الاستفسار البنائية. ويوضح الشكل رقم (٧-١) اللغات الفرعية (أو الفئات) الثلاث المكونة للغة الاستفسار البنائية.

شكل رقم (٧-١): اللغات الفرعية (أو الفئات الثلاث من التعليمات) المكونة للغة الاستفسار البنائية



ونظراً لأهمية لغة الاستفسار البنائية فى قواعد البيانات العلاقية، فإن هذا يستلزم شرح مكوناتها الأساسية شرحاً تطبيقياً مستفيضاً؛ حتى يتسنى فهم طريقة عمل تعليماتها. لذا فقد تم تخصيص هذا الفصل والفصل التالى لشرح المكونات الأساسية للغة الاستفسار البنائية. ويحتوى هذا الفصل على شرح لتعليمات لغة تعريف (هياكل) البيانات، وعلى شرح للعبارات الأساسية فى تعليمة الاختيار التى تعد من أهم تعليمات لغة معالجة البيانات. أما الفصل التالى فيستكمل شرح مكونات لغة الاستفسار البنائية بحيث خصص الجزء الأول منه لاستكمال شرح تعليمة الاختيار، عندما تقوم التعليمة بالتعامل مع أكثر من جدول فى آن واحد، ولشرح بقية تعليمات لغة معالجة البيانات. أما الجزء الثانى من الفصل الثامن فقد خصص لشرح تعليمات لغة التحكم فى البيانات.

وسنستخدم فى هذا الفصل والفصل الثامن نظام إدارة قاعدة بيانات أوراكل فى تنفيذ واختبار تعليمات لغة الاستفسار البنائية القياسية، وذلك لكون هذه البيئة واحدة من الأوسع انتشاراً بين المتخصصين فى تطوير التطبيقات المبنية على نظم قواعد البيانات، هذا بالإضافة لتشابهها مع ما توفره نظم إدارة قواعد البيانات المعروفة الأخرى على المستوى التجارى مثل «دى بى ٢» (DB2) و«سايبيس» (SYBASE) من بيئات مشابهة لتنفيذ تعليمات لغة الاستفسار البنائية. أما بالنسبة لمن لا تتوافر لديهم بيئة أوراكل، وحتى يتمكن هؤلاء من تطبيق (بعض) مفاهيم وتعليمات لغة الاستفسار

البنائية والاستفادة القصوى من محتويات هذين الفصلين، فنستخدم قاعدة بيانات أكسس (ACCESS)، وبشكل مقتضب: وذلك لتوافرها فى غالبية الحاسبات الشخصية فى وقتنا الراهن.

٧-١ لغة تعريف البيانات ((Data Definition Language (DDL)):

٧-١-١ تعليمة الإنشاء (Create Statement):

تحتوى لغة الاستفسار البنائية التى تم وضع مقاييسها عام ١٩٩٢ (SQL-92) فى مستواها المبدئى على ثلاثة أنواع من تعليمات الإنشاء: إنشاء قاعدة بيانات، وإنشاء جدول، وإنشاء منظور. بالإضافة إلى ذلك فإن مقاييس لغة الاستفسار البنائية (SQL-92) تحتوى على خمسة أنواع أخرى من تعليمات الإنشاء فى مستواها المتوسط، من ضمنها تعليمة إنشاء القيود (Create Assertion)، وتعليمة إنشاء مدى من القيم (Create Domain). ويلاحظ أن تعليمة إنشاء فهرس (Create Index) قد تم حذفها من هذا المقياس على الرغم من تضمينها فى مقاييس سابقة، وذلك لكون الفهارس تتعلق بعمليات تحسين أداء قاعدة البيانات ولا تعد جزءاً من عمليات إنشاء قواعد البيانات العلاقية أو تداول محتوياتها. وفيما يلى شرح لتعليمات إنشاء قاعدة بيانات، وإنشاء جدول، وإنشاء مدى، وإنشاء منظور، وإنشاء فهرس، على التوالى.

٧-١-١-١ إنشاء قاعدة بيانات (Create Schema):

لم تحتوِ المقاييس السابقة لمقياس (١٩٩٢) على مبدأ تعريف هياكل لأكثر من قاعدة بيانات واحدة وإنما كانت جميع تعاريف العلاقات والمكونات الأخرى جزءاً لهيكل واحد لقاعدة البيانات. إلا أن مقياس (١٩٩٢) أوجد مبدأ هيكل قاعدة البيانات الذى يتم من خلاله تجميع جميع العلاقات المرتبطة فيما بينها والمكونات الأخرى لقاعدة البيانات الواحدة ضمن هيكل واحد. ويعرف هيكل قاعدة البيانات من خلال استخدام التعليمة التالية:

```
CREATE SCHEMA Database_Name AUTHORIZATION Owner_UserID;
```

وتمثل الكلمات المكتوبة بالأحرف الكبيرة وبالخط الداكن كلمات محجوزة يجب أن تتضمنها تعليمة إنشاء هيكل قاعدة البيانات. أما الكلمة (Database_Name) فتمثل اسم

هيكل قاعدة البيانات قيد الإنشاء، والكلمة (Owner_UserID) تمثل رمز دخول المستخدم الذى سيكون مالكاً لهيكل قاعدة البيانات والذى من حقه التصرف بمحتوياتها وإعطاء الصلاحيات عليها. وعادة يتم استخدام تعليمة إنشاء هيكل قاعدة البيانات السابقة من قبل إدارى قاعدة البيانات فى المنشأة فقط. ولا شك أن مبدأ استخدام تعريف هياكل قواعد البيانات مفيد جداً فى الغالبية العظمى من المنظمات. فعلى سبيل المثال، يتم عادة استخدام هيكلين من قواعد البيانات فى غالبية المنظمات التى تقوم بتطوير النظم التطبيقية التى تتعامل مع قاعدة البيانات. أحد هذين الهيكلين مخصص لاختبار التطبيقات وقاعدة البيانات (Testing Environment) والثانى مخصص للنظم العاملة فعلياً فى المنظمة (Production Environment).

ومثال على البيئة التى تمت الإشارة إليها أعلاه، يمكن تعريف قاعدة البيانات الاختبارية كما يلى:

```
CREATE SCHEMA Company_Testing AUTHORIZATION Saleh_Ahmed;
```

حيث إن اسم هيكل قاعدة البيانات الاختبارية هو (Company_Testing) وأن الشخص صاحب الصلاحية الكاملة عليها هو «صالح أحمد» الذى لديه رمز المستخدم (Saleh_Ahmed). ويمكن إنشاء هيكل قاعدة البيانات فى البيئة التشغيلية (أو الإنتاجية) بالطريقة نفسها وكما يلى:

```
CREATE SCHEMA Company AUTHORIZATION Mohammed_Ali;
```

وبعد إنشاء هيكلى قاعدتى البيانات يمكن إنشاء هياكل العلاقات والمكونات الأخرى لكل منها، وبحيث يمكن أن تتطابق مكونات كلا الهيكلين بما فى ذلك مسميات تلك المكونات فى كلا الهيكلين. ويعنى هذا أن مقياس ١٩٩٢ يمكّن من إنشاء أكثر من هيكل عوضاً عن هيكل واحد فقط يحتوى على مكونات كلا الهيكلين السابقين. ويزود هذا المبدأ إداريى قواعد البيانات والمستفيدين منها بمرونة كبيرة. فعلى سبيل المثال، يمكن تحويل قاعدة البيانات الاختبارية إلى البيئة التشغيلية بمجرد تغيير مسمى هيكل قاعدة البيانات وتزويدها ببيانات المنظمة الموجودة أصلاً فى البيئة التشغيلية السابقة دون الحاجة إلى تغيير مسميات أو مكونات أى من العلاقات والمكونات الأخرى فى هيكل قاعدة البيانات الاختبارية. ومثل هذه العملية لا يمكن أن تتم بهذه البساطة دون مبدأ تعريف هياكل قواعد البيانات إذ إن كل مكون لها فى البيئة التشغيلية يجب

أن يأخذ اســماً مختلفاً عن البيئة الاختباريــة. ويتفاقم هذا الوضع ويتعقد فى حالة وجــود العديد من المكونات التابعة منطقياً لهيــاكل مختلفة من قواعد البيانات ولكنها معرفة ضمن هيكل واحد.

٧-١-١-٢ إنشاء جدول (Create Table):

تســتخدم تعليمة إنشــاء جدول (Create Table) لتعريف علاقة جديدة ضمن هيكل قاعدة البيانات. وتتضمن التعليمة اســم العلاقــة وتعريفاً للحقول التى تحتويها وأية قيود مفروضة عليها، كما يوضح الشكل العام التالى للتعليمة:

```
CREATE TABLE TableName
(Column-Definition [,Column-Definition]);
```

حيث إن (Column-Definition) فى الشكل العام للتعليمة يمثل تعريفاً للحقول المكونة للعلاقة، وأن كل حقل يجب أن يرتبط بمســمى معين وبنوع واحد من أنواع القيم التى يكمــن أن يحتــوى عليها وأية قيــود مفروضة على الحقل كما هو موضح فى الشــكل التالى:

```
ColumnName    Data-Type    [Constraint]
```

كما يمكن إضافة المفتاح الرئيســى، والمفاتيح الخارجية التى تمثل قيود الســلامة المرجعية ضمن تعليمة الإنشاء وبعد تعريف حقول العلاقة، أو تعريفها لاحقاً من خلال استخدام تعلمية تغيير (أو تعديل) العلاقة (Alter Table).

ومن القواعد التى يجب مراعاتها عند تعريف أية علاقة ما يلى:

- يجب ألا يتكرر اسم (Column Name) أى حقل فى الجدول نفسه.
- يجب أن يكون نوع بيانات (Data Type) أى حقل من ضمن الأنواع المعروفة لقاعدة البيانات (مثل السلاسل الحرفية، الأرقام الصحيحة، إلخ) أو الأنواع التى يقوم مصمم قاعدة البيانات بتعريفها.
- توفر لغة الاستفســار البنائية عدة أنواع من القيود (Constraints) التى يمكن أن تفرض على الأعمدة منها (Not Null) الذى يقصد منه أنه لا بد أن يكون للحقل قيمــة معرفة، و(Unique) الذى يقصد منه أن قيمة الحقل يجب أن تكون فريدة

تمايز بين كافة السجلات المخزنة فى العلاقة، و(Primary Key) الذى يقصد منه أن الحقل هو المفتاح الرئيسى للعلاقة. وعند فرض قيد المفتاح الرئيسى على حقل ما فإن هذا يعنى ضمنياً أن الحقل (Unique) ولا يمكن أن تكون قيمةُ غير معرفة (Null). بالإضافة إلى ذلك يوجد أنواع أخرى من القيود التى سوف نتطرق إليها لاحقاً فى هذا الفصل.

ومثال على تعليمة إنشاء جدول، لنفترض أننا نرغب فى إنشاء جدول الأقسام العلمية فى الجامعة الأهلية بمسمى (Department_T) الذى يكمن إنشاؤه حسب تعليمة الإنشاء التالية:

```
CREATE TABLE DEPARTMENT_T
   (DEPARTMENT_ID      CHAR (6)      NOT NULL,
   NAME                CHAR (30)     NOT NULL,
   CONSTRAINT DEPARTMENT_PK PRIMARY KEY (DEPARTMENT_ID));
```

يتكون تعريف الجدول السابق من حقلين هما رمز القسم (Department_ID)، واسم القسم (Name) وأن نوعية بيانات كل منهما عبارة عن سلسلة حرفية (Char) إلا أن طول السلسلة الحرفية للحقل الأول هى ستة أحرف، فى حين أن طول سلسلة الثانى الحرفية ثلاثون حرفاً. كما يحتوى تعريف الجدول على ثلاثة قيود اثنان منها يتعلقان بالحقل الأول والحقل الثانى حيث إن كلا الحقلين يجب أن لا يحتويا على قيم غير معرفة، فى حين أن القيد الثالث يبين المفتاح الرئيسى للجدول، وقد سمى هذا القيد بمسمى (Department_PK). ويعنى هذا القيد أن المفتاح الرئيسى للجدول هو رمز القسم (Department_ID) وأن مسماه هو (Department_PK). ويعد تزويد القيود بمسميات من الأساليب الحميدة التى يتبعها إداريو قواعد البيانات عند إنشائهم لقواعد البيانات حيث يمكن هذا الأسلوب من التعرف على القيود المفروضة على قاعدة البيانات فى وقت لاحق من خلال استعراضهم لكتلوج النظام. كما يمكنهم أيضاً من إلغاء القيود وإعادة فرض قيود أخرى دون الحاجة إلى حذف الجدول (وكافة محتوياته) ومن ثم إعادة تعريفه بقيود جديدة (أو معدلة). وفى حالة عدم الرغبة فى إنشاء قيد المفتاح الرئيسى بمسمى معين فإنه يمكن كتابة القيد كما يلى:

```
CREATE TABLE DEPARTMENT_T
    (DEPARTMENT_ID     CHAR (6)    NOT NULL,
    NAME               CHAR (30)   NOT NULL,
    PRIMARY KEY (DEPARTMENT_ID));
```

أو يمكن تعريف المفتاح الرئيسى مباشرة كقيد على رمز القسم كما يلى:

```
CREATE TABLE DEPARTMENT_T
    (DEPARTMENT_ID     CHAR (6)    PRIMARY KEY,
    NAME               CHAR (30)   NOT NULL);
```

٧-١-١-٢-١ أنواع البيانات (Data Types):

توفر لغة الاستفسار البنائية بعض أنواع البيانات الأساسية، المضمنة فى مقاييس اللغة، إلا أن غالبية نظم إدارة قواعد البيانات التجارية توفر أنواع بيانات إضافية لا تتضمنها مقاييس اللغة. ولذلك فإننا نجد اختلافاً فيما توفره نظم قواعد البيانات التجارية من أنواع بيانات إضافية. ويأتى من ضمن أنواع البيانات الأساسية التى تتضمنها لغة الاستفسار البنائية القياسية الأرقام، والسلاسل الحرفية، والسلاسل المكونة من الأرقام الثنائية (Bits)، والقيم المنطقية الثنائية (Boolean)، والتاريخ، والوقت. وفيما يلى إيضاح لأنواع البيانات الأساسية فى لغة الاستفسار البنائية.

- **الأرقام (Numeric):** تتكون أنواع الأرقام من الأعداد الصحيحة (INTEGER) والأرقام الصحيحة القصيرة (SMALLINT) والأعداد الحقيقية (REAL) بمقاسات مختلفة من الدقة (Precision) حيث يمكن استخدام Decimal(i,j) أو Number(i,j) بحيث يدل (i) على العدد الكلى للخانات العشرية (أو درجة الدقة)، على حين يدل (j) على عدد الخانات العشرية بعد الفاصلة العشرية. وتكون القيمة الافتراضية لعدد الخانات بعد الفاصلة العشرية فى حالة عدم تعريفها صفراً.
- **السلاسل الحرفية (Character-String):** يمكن تعريف السلاسل الحرفية على أنها ثابتة أو متغيرة الطول. وتعرف السلاسل الحرفية ثابتة الطول بـ ((Char (n)) أو ((Character (n)) بحيث يرمز (n) للعدد الأكبر من الحروف التى من الممكن أن تحتويها السلسلة الحرفية. وفى حالة قلة عدد حروف السلسلة الحرفية للحقل عن العدد الأكبر فإنه يتم إضافة حروف فاضية (Blank Characters) لاستكمال السلسلة الحرفية

حتى تصل للعدد (n). فعلى سبيل المثال، عند إدخال الاسم (Ahmed) فى حقل الاسم لعلاقة ما فإنه سيخزن على أساس '   Ahmed' بحيث يتم إضافة خمس حروف فاضية لاستكمال السلسلة الحرفية على افتراض تعريفها بأنها بطول عشرة أحرف. كما يجب استخدام علامتى التنصيص المفردة أثناء عملية إدخال البيانات النصية. فعند إدخال الاسم (Ahmed) يجب أن يدخل 'Ahmed'.

أما بالنسبة للسلاسل الحرفية متغيرة الطول فتعرف بـ VCHAR (n) أو VARYING (n) أو CHARACTER VARYING (n) أو VARCHAR (n). ومرة أخرى يرمز 'n' للعدد الأكبر من الحروف التى من الممكن أن تحتويها السلسلة الحرفية. ويكمن وجه الاختلاف بين السلسلة الحرفية الثابتة الطول والسلسلة الحرفية المتغيرة الطول فى كون الأخيرة تحتوى على العدد الفعلى للبيانات المدخلة دون إضافة حروف فاضية مما ينتج عنه سجلات تابعة لنفس العلاقة ولكن بمقاسات مختلفة عوضاً عن كونها ثابتة المقاسات. كما تجدر الإشارة إلى أن السلاسل الحرفية حساسة لأحجام الحروف فالحرف "c" يعد مختلفاً عن الحرف "C" على سبيل المثال. ويعنى هذا أن الاسم 'Ahmed' مختلف عن الاسم 'AHMAD' والاسم 'AHmed'.

- **سلاسل الأرقام الثنائية (Bit-string)**: يمكن أن تكون سلاسل الأرقام الثنائية إما ثابتة الطول بحيث تعرف على أساس BIT(n)، وإما متغير الطول وتعرف على أساس BIT VARYING (n). ويقصد بـ (n) فى كلتا الحالتين الحد الأقصى لطول سلسلة الأرقام الثنائية. والحالة الافتراضية عند عدم تحديد الحد الأقصى يكون سلسلة رقمية ثنائية بطول واحد ((BIT (1)). وللتفريق بين السلاسل الحرفية والسلاسل الرقمية، فإنه يجب تحديد الحرف "B" قبل السلسلة الرقمية الثنائية مثل B '10101100'.
- **القيم المنطقية الثنائية (Boolean)**: القيمة المنطقية الثنائية، وكما هو متعارف عليه فى لغات البرمجة الأخرى، من الممكن إما أن تكون «صح» (True) أو أن تكون «خطأ» (False). إلا أنه بسبب سماح لغة الاستفسار البنائية باستخدام القيمة غير المعرفة (Null)، فإن هذا يستدعى استخدام القيمة الثالثة غير المعلومة (UNKNOWN) ضمن ما يعرف بالمنطق الثلاثى القيم (True, False, Unknown)، الذى سنتطرق له لاحقاً (فى الجزء ٧-٢-١-٥-٢-١).
- **التاريخ (Date)**: نوع بيانات التاريخ يتكون من عشر خانات مقسمة إلى السنة، والشهر، واليوم يفصل بينها علامة الناقص (-) (YYYY-MM-DD)، بحيث يقصد بالـ (YYYY) السنة، و(MM) الشهر، و(DD) اليوم.

- **الوقت (Time):** نوع بيانات الوقت يتكون من ثمانى خانات (على الأقل) تمثل الساعة، والدقيقة، والثانية وعلى هيئة (HH:MM:SS) بحيث يقصد بـ (HH) الساعة (Hour)، و(MM) الدقيقة (Minute)، و(SS) الثانية (Second).

٧-١-١-٣ توصيف القيود فى لغة الاستفسار البنائية والتعامل معها:

توفر لغة الاستفسار البنائية عدداً من أنواع القيود التى يمكن توصيفها على قاعدة البيانات التى يجب على نظام إدارة قاعدة البيانات التأكد من تحققها على حالات قاعدة البيانات كافة. ويمكن تقسيم أنواع القيود التى يمكن توصيفها بلغة الاستفسار البنائية إلى أربعة أنواع رئيسة هى: قيود الحقول والمدى (Attribute and Domain Constraints)، قيود المفاتيح الرئيسة والسلامة المرجعية (Key and Referential Integrity Constraints)، وقيود السجلات (Tuple Constraints)، والقيود العامة على قاعدة البيانات (Assertions).

٧-١-١-٣-١ قيود الحقول والمدى (Attribute and Domain Constraints):

تسمح لغة الاستفسار البنائية بأن تأخذ قيم حقول الجداول القيمة غير المعرفة (Null) فى حالة عدم إدخالها فى هذه الحقول. وللتأكد من أن القيم المدخلة فى حقل ما يجب أن تكون معرفة مثل الاسم الأول واسم العائلة للموظفين، التى لا يمكن أن تكون غير معرفة منطقياً، فإن لغة الاستفسار البنائية توفر قيد (Not Null) الذى يمكن توصيفه على مثل هذه الحقول. ويعنى هذا أن قيم هذه الحقول يجب أن تتوافر فى جميع سجلات الموظفين، ولا يمكن أن تكون غير معرفة.

توفر لغة الاستفسار البنائية أيضاً القيد الفريد (Unique) الذى يمكن فرضه على أحد الحقول فى حالة كانت قيمة هذا الحقل يجب ان تكون فريدة لا تتكرر مع بقية السجلات فى الجدول نفسه. فعلى سبيل المثال، يمكن فرض هذا القيد على رقم السجل المدنى للموظفين (أو رقم بطاقة الأحوال المدنية) لكون قيم مثل هذا الحقل لا يمكن أن تتكرر بين الموظفين. وعلى الرغم من أن قيمة الحقل المعرف على أساس أنه فريد، من المنطقى أن لا يمكن أن يأخذ القيمة غير المعرفة (Null)، إلا أن SQL تسمح بذلك. والسبب وراء ذلك يعزى إلى أن كل قيمة غير معرفة تختلف عن قيمة غير معرفة أخرى للحقل نفسه، بمعنى أن (Null) فى حقل رقم الهاتف لأحد الموظفين تختلف عن القيمة (Null) فى الحقل نفسه ولكن لموظف آخر.

أمـا النـوع الثالث من قيود الحقول التى توفرها لغة الاستفسـار البنائية فهو قيد المفتـاح الرئيسـى (Primary Key). ويعنى هذا القيد عنـد توصيفه على أحد الحقول فى جدول ما، أن هذا الحقل هو المفتاح الرئيسـى للجدول، كما سـبق أن أشرنا لذلك سابقاً. وعند توصيف حقل ما على أنه مفتاح رئيسي، فإن ذلك يعنى ضمنياً أن الحقل فريـد ولا يمكن أن يكون غير معرف، بمعنى أن كلا القيدين السـابقين يتحققان على الحقل عند تعريفه باعتباره مفتاحاً رئيسياً للجدول.

وفيمـا يلى الصيغة العامة لتوصيف قيود الحقـول بحيث يقصد بالعلامة (|) كلمة «أو» فـى حين يقصد بالكلمـات الواقعة ضمن أقواس مربعة بأنهـا كلمات اختيارية. ولكـون كافة قيود الحقول تقع ضمن قوسـين مربعين فإن هذا يعنى أن توصيف قيود علـى الحقول عملية اختيارية، وأن عدم وضع أى قيدٍ مـن القيود الثلاثة أعلاه يعنى أن الحقل قد يحتوى على قيم غير معرفة فى بعض (أو كل) سجلاته، وأن قيم الحقل ليست فريدة، وأن الحقل ليس مفتاحاً رئيسياً للجدول.

```
ColumnName    Data-Type    [[Not] Null | Unique | Primary Key]
```

وتوضح تعليمة الإنشـاء التالية، المسـتمدة من قاعدة بيانات الجامعة الأهلية، قيد المفتاح الرئيسى وقيد القيمة المعرفة (Not Null)، حيث يبين أن المفتاح الرئيسى لجدول المواد الدراسية هو رمز المادة الذى يتكون من سلسلة حرفية بمقاس ثابت طوله سبعة أحرف، كما يوضح أن كلاً من حقل اسـم المادة الدراسـية (Title) وحقل عدد وحداتها الدراسية (Units) لا يمكن أن يكونا غير معرفين.

```
CREATE TABLE COURSE_T
      (COURSE_ID      CHAR (7)      PRIMARY KEY,
      TITLE           CHAR (35)     NOT NULL,
      UNITS           NUMBER        NOT NULL);
```

كمـا يمكن أيضاً فرض قيود على مدى القيـم التى من الممكن أن يأخذها حقل ما باسـتخدام كلمـة التحقق (CHECK). فعلى سـبيل المثال، من الممكـن أن يفرض قيد التحقق على عدد وحدات (أو سـاعات) المادة الدراسـية، بحيث يكون أكبر من صفر وأقل من ست وحدات دراسية، كما يوضح المثال التالى:

```
CREATE TABLE COURSE_T
    (COURSE_ID  CHAR (7)     PRIMARY KEY,
    TITLE       CHAR (35)    NOT NULL,
    UNITS       NUMBER       NOT NULL CHECK (UNITS > 0 AND UNITS < 6));
```

٧-١-١-٣-١-١ إنشاء المدى (Domain Creation):

على الرغم من أنه يمكن تعريف نوع البيانات لأى حقل بشكل مباشر، إلا أنه يمكن أيضاً تعريف مدى معين لأحد أنواع البيانات واستخدام هذا المدى عند توصيف بعض الحقول. وتستخدم عبارة إنشاء مدى (Create Domain) لإنشاء نوع جديد من البيانات مشتق من أحد أنواع البيانات الرئيسية التى توفرها لغة الاستفسار البنائية. وتعد هذه الطريقة مفيدة جداً عندما يشترك عدد من الحقول فى المدى نفسه من نوعية البيانات. فعلى سبيل المثال، يمكن تعريف رقم السجل المدنى (أو رقم بطاقة الأحوال المدنية) على أنها مدى يتكون من سلسلة حرفية مكونة من عشرة حروف كما يلى:

```
CREATE DOMAIN Social_Identification_Number AS CHAR (10);
```

وعند توصيف أى حقل يتعلق برقم السجل المدنى يستخدم اسم المدى (Social_Identification_Number) عوضاً عن توصيفه باستخدام سلسلة حرفية مكونة من عشرة أحرف ((Char (10)). وتعد هذه الطريقة فى توصيف الحقول مفيدة جداً لكونها تسهل قراءة مكونات قاعدة البيانات من جانب، وإمكانية حصر التغييرات فى مكان واحد فقط من جانب آخر. فعلى سبيل المثال، يمكن تغيير تعريف المدى الخاص برقم السجل المدنى ليصبح بطول اثنى عشر حرفاً عوضاً عن عشرة أحرف، وسينعكس هذا التغيير على كل الحقول التى تم توصيف نوع بياناتها على أنه من نوع (Social_Identification_Number).

ومثال آخر، يمكن تعريف مدى باسم «عدد الوحدات الدراسية» (CRS_UNITS) بحيث يكون من نوع بيانات الأعداد وفى الوقت نفسه يفرض عليه قيد التحقق الذى ينص على أن عدد الوحدات الدراسية يجب أن يكون أكبر من صفر وأقل من ست وحدات، كما يلى:

```
CREATE DOMAIN CRS_UNITS AS NUMBER
        CHECK (CRS_UNITS> 0 AND CRS_UNITS < 6);
```

ويمكننا اســتخدام المدى الذى تم إنشــاؤه كنوع لبيانــات (Data Type) أى حقل فى قاعــدة البيانات يعرف على أســاس أنه من مــدى CRS_UNITS، وبحيث يطبق عليه قيد التحقق الذى ينص على أن عدد وحدات المادة الدراســية يجب أن يكون أكبر من صفر وأقل من ســت وحدات. فمثلاً يمكن إعادة تعريف جدول المواد الدراسية بحيث يستخدم المدى الذى تم تعريفه ليصبح كالتالى:

```
CREATE TABLE COURSE_T
    (COURSE_ID    CHAR (7)      PRIMARY KEY,
     TITLE        CHAR (35)     NOT NULL,
     UNITS        CRS_UNITS     NOT NULL);
```

أما المثال التالى فيوضح تعريف مدى الجنس (Gender) الذى وضع عليه قيد التحقق بحيث تكون القيمة المدخلة لأى حقل يستخدمه نوعاً لبياناته بأن تكون إما ذكراً (Male) أو أنثــى (Female)، وبحيث يســتخدم الحرف الأول فقط من جنس الشــخص (M) أو (F).

```
CREATE DOMAIN GENDER AS CHAR (1)
               CHECK (VALUE IN ('M', 'F'));
```

تجدر الإشارة هنا إلى أن إنشاء مدى يعد من ضمن مقياس (SQL-92)، ولكنه ليس مــن الضرورة لنظام إدارة قاعدة البيانات أن يتبنى هذا الجزء من المقياس حتى يكون متوافقاً معه فى المستوى المبدئى أو المتوسط.

من الممكن أيضاً أن يتم تعريف قيمة افتراضية لحقل ما تســتخدم من قبل النظام فــى حالة عدم إدخال قيمة للحقل فى أثناء عملية إدخال البيانات. وتســتخدم كلمة (DEFAULT) فى مقياس (SQL-92) لتعريف القيمة الافتراضية الواجب إدخالها تلقائياً مــن قبل نظام إدارة قاعدة البيانات فى حالة عدم إدخال قيمة للحقل. ويوضح المثال التالى طريقة اســتخدام القيمة الافتراضية فى جدول المواد الدراســية، بحيث تكون القيمــة الافتراضية لعدد وحدات المادة الدراســية «ثلاثة» فى حالة عدم إدخال قيمة لحقل عدد الوحدات الدراسية.

```
CREATE TABLE COURSE_T
    (COURSE_ID    CHAR (7)      PRIMARY KEY,
     TITLE        CHAR (35)     NOT NULL,
     UNITS        CRS_UNITS     NOT NULL DEFAULT 3);
```

٧-١-١-٣-٢ قيود المفاتيح الرئيسية والسلامة المرجعية (Key and Referential Integrity Constraints):

تستخدم عبارة المفتاح الرئيسى (Primary Key) التى توفرها (SQL) لتعريف المفاتيح الرئيسية للجداول. أما المفاتيح الخارجية فيتم تعريفها من خلال عبارة مفتاح خارجى (Foreign Key). وعلى الرغم من أن المفتاح الرئيسى يمكن تعريفه باعتباره قيداً على حقل ما فى الجدول، فى أثناء تعريف الحقل، كما سبق إيضاح ذلك فى مثال جدول المواد الدراسية أعلاه، إلا أن المفتاح الرئيسى يجب أن يعرف بشكل مستقل عن تعريف حقول الجدول عندما يكون المفتاح الرئيسى مكوناً من أكثر من حقل. ويوضح المثال التالى الذى يعرف جدول المجموعات الدراسية فى الجامعة الأهلية أن المفتاح الرئيسى يتكون من أربعة حقول هى: رمز المادة الدراسية (Course_ID)، ورقم الشعبة (أو المجموعة) (Section_No)، والفصل الدراسى المنفذة فيه (Semester)، والسنة الدراسية المنفذة فيها (Year). كما يحتوى تعريف الجدول على مفتاحين خارجيين: الأول منهما يربط المجموعة الدراسية بالمادة الدراسية التى تتبعها فى جدول المواد الدراسية، أما الثانى فيربط المجموعة الدراسية بعضو هيئة التدريس الذى يقوم بتدريسها.

```
CREATE TABLE SECTION_T
   (COURSE_ID          CHAR (7)        NOT NULL,
   SECTION_NO          NUMBER          NOT NULL,
   SEMESTER            CHAR (10)       NOT NULL,
   YEAR                NUMBER          NOT NULL,
   FACULTY_ID          CHAR (10)       NOT NULL,
   PRIMARY KEY (COURSE_ID, SECTION_NO, SEMESTER, YEAR),
   FOREIGN KEY (COURSE_ID)
               REFERENCES COURSE_T (COURSE_ID),
   FOREIGN KEY (FACULTY_ID)
               REFERENCES FACULTY_T (FACULTY_ID));
```

تجدر الملاحظة أنه ليس من الضرورى أن يتطابق اسم الحقل باعتباره مفتاحاً خارجياً فى جدول ما مع اسم الحقل الذى يشير إليه فى الجدول الآخر، ولكنه من الضرورى أن يكون كلا الحقلين من نوعية البيانات نفسها. فمثلاً يمكن تسمية حقل رمز عضو هيئة التدريس على أنه (FAC_ID) فى جدول المجموعات الدراسية دون أن يغير ذلك فى الأمر من شىء ما دامت نوعية بيانات الحقل هى من نوعية بيانات الحقل (Faculty_ID) نفسها فى جدول أعضاء هيئة التدريس، وذلك كما يلى:

```
CREATE TABLE SECTION_T
   (COURSE_ID          CHAR (7)       NOT NULL,
   SECTION_NO          NUMBER         NOT NULL,
   SEMESTER            CHAR (10)      NOT NULL,
   YEAR                NUMBER         NOT NULL,
   FAC_ID              CHAR (8)       NOT NULL,
   PRIMARY KEY (COURSE_ID, SECTION_NO, SEMESTER, YEAR),
   FOREIGN KEY (COURSE_ID)
               REFERENCES COURSE_T (COURSE_ID),
   FOREIGN KEY (FAC_ID)
               REFERENCES FACULTY_T (FACULTY_ID));
```

أمــا فى حالة تطابق مســمى الحقلين فى كلا الجدولين فإنه بالإمكان الاســتغناء عــن ذكر الحقل المشــار إليه من قبل المفتاح الخارجى كما يوضــح المثال التالى، حيث تم الاســتغناء عن ذكر اسم الحقل (Course_ID) والحقل (Faculty_ID) عند تعريف كلا المفتاحين الخارجيين لكون مســمياتهما فى جدول المجموعات الدراســية متوافقة مع مسمياتهما فى جدول المواد الدراسية وجدول أعضاء هيئة التدريس، على التوالى:

```
CREATE TABLE SECTION_T
   (COURSE_ID          CHAR (7)       NOT NULL,
   SECTION_NO          NUMBER         NOT NULL,
   SEMESTER            CHAR (10)      NOT NULL,
   YEAR                NUMBER         NOT NULL,
   FACULTY_ID          CHAR (8)       NOT NULL,
   PRIMARY KEY (COURSE_ID, SECTION_NO, SEMESTER, YEAR),
   FOREIGN KEY (COURSE_ID)
               REFERENCES COURSE_T,
   FOREIGN KEY (FACULTY_ID)
               REFERENCES FACULTY_T);
```

ويقوم نظام إدارة قاعدة البيانات بإعطاء كل قيد رمزاً يميزه عن بقية القيود المفروضة علــى قاعدة البيانــات. إلا أنه من المتعارف عليه عند إداريى قواعد البيانات إعطاء كل قيد مســماه الخاص الذى يساعد على فهمهم لطبيعة القيد، (سواء كان مفتاحاً رئيسياً أم خارجياً أم غير ذلك) ومجال تطبيقه (سواء كان على حقل أم جدول أم قيد عام). كما يساعدهم ذلك على التعرف على القيود المختلفة وتعطيل العمل بها أو تعديلها. ويوضح المثال التالى إحدى الطرق المتبعة عند تسمية القيود، بحيث تم إدراج اسم الجدول ضمن مسمى القيد وطبيعة كونه مفتاحاً رئيسياً (PK) أو خارجياً (FK).

```
CREATE TABLE SECTION_T
   (COURSE_ID          CHAR (7)      NOT NULL,
   SECTION_NO          NUMBER        NOT NULL,
   SEMESTER            CHAR (10)     NOT NULL,
   YEAR                NUMBER        NOT NULL,
   FACULTY_ID          CHAR (8)      NOT NULL,
   CONSTRAINT SECTION_PK PRIMARY KEY (COURSE_ID,
   SECTION_NO, SEMESTER, YEAR),
   CONSTRAINT SECTION_FK1 FOREIGN KEY (COURSE_ID)
              REFERENCES COURSE_T,
   CONSTRAINT SECTION_FK2 FOREIGN KEY (FACULTY_ID)
              REFERENCES FACULTY_T);
```

ولكــون المفاتيح الخارجية هى الطريقــة الوحيدة لتمثيــل العلاقات بين الحالات المدونة فى الجداول العلاقية، فإنها تمثل قيوداً للسلامة المرجعية. فعلى سبيل المثال، عندما نقول إن «كل مجموعة دراســية تتبع لمادة دراســية واحدة»، فإن هذا يعد قيداً بين المجموعة الدراســية والمادة الدراســية التى تتبعها المجموعة. وتمثل هذه العلاقة مــن خلال تعريف مفتاح خارجى فى جدول المجموعات الدراســية يشــير إلى المفتاح الرئيسى فى جدول المواد الدراسية، كما أسلفنا فى المثال أعلاه. إلا أن السؤال المتعلق بقيد السلامة المرجعية هو ماذا يحصل لو تغير رمز المادة الدراسية التى تتبعها إحدى المجموعات الدراســية، أو حذفت مادة دراسية من جدول المواد الدراسية ويتبعها عدد من المجموعات فى جدول المجموعات الدراســية؟ فى هذه الحالة، ما مصير المفاتيح الخارجية؟ إلام تشير هذه المفاتيح؟

كذلك هو الحال عند إضافة مجموعة دراســية دون تحديد لرمز المادة الدراســية التى تتبعها، أو تم إدخال المجموعة الدراســية بقيمة لرمز مادة دراســية غير موجودة أساساً فى جدول المواد الدراسية، أو عدل رمز المادة الدراسية التى تتبعها مجموعة ما ليشير لمادة غير موجودة فى جدول المواد الدراسية. ما نتيجة مثل هذه العمليات؟

إن ردة الفعل الافتراضية فى نظم قواعد البيانات العلاقية حسب مقياس (SQL-92) هو رفض مثل هذه العمليات دون تغيير لمحتويات جداول قاعدة البيانات، وذلك لكونها تؤدى لاختراق قيود الســلامة المرجعية. وتســمى ردة الفعل هذه بعملية منع أو إيقاف التنفيــذ (Restrict). ولكن مقياس (SQL-92) يوفــر ثلاثة بدائل أخرى لمصممى قواعد البيانات، بالإضافة إلى ردة الفعل الافتراضية، تمكنهم من اختيار الفعل المناســب عند

اختراق قيود السلامة المرجعية حسب الوضع الذى يتناسب مع قواعد بياناتهم. ويتم ذلك من خلال ربط الفعل المناسب بقيد المفتاح الخارجى. أما ردود الفعل الثلاثة فهى: وضع المفتاح الخارجى فى حالة غير معرفة (Set Null)، التغيير المتسلسل (Cascade)، وضع المفتاح الخارجى فى الحالة الافتراضية (Set Default). وعند اختيار أحد ردود الفعل المناسبة فإنه يجب أن يرتبط بالفعل نفسه، سواء كان فعل تعديل (On Update) أم فعل حذف (On Delete). ولقد سبق شرح مفاهيم قيود السلامة المرجعية فى الجزء ٤-١-٤-٤ والتعامل مع اختراقاتها من قبل نظام إدارة قاعدة البيانات فى الجزء ٤-١-١-٤-٥.

ويوضح المثال التالى نوعين من ردود الفعل: الأول منهما ينص على أن تحديث رمز المادة الدراسية فى جدول المواد الدراسية يجب أن ينعكس على (أو يتسلسل إلى) جميع مجموعات المادة الدراسية فى جدول المجموعات الدراسية، وكذلك هو الحال بالنسبة لتحديث رمز عضو هيئة التدريس الذى يجب أن ينعكس على (أو يتسلسل إلى) جميع السجلات التى يدرِّسها عضو هيئة التدريس نفسه فى جدول المجموعات الدراسية. أما رد الفعل الثانى فيتمثل فى حالة إلغاء سجل أحد أعضاء هيئة التدريس من جدول أعضاء هيئة التدريس. فى هذه الحالة سيتم تغيير قيمة حقل رمز عضو هيئة التدريس بحيث يأخذ القيمة الافتراضية وهى (No Body) فى كل سجل من سجلات جدول المجموعات الدراسية التى تحتوى على قيمة رمز عضو هيئة التدريس الذى تم إلغاؤه من جدول أعضاء هيئة التدريس.

```
CREATE TABLE SECTION_T
    (COURSE_ID          CHAR (7)        NOT NULL,
    SECTION_NO          NUMBER          NOT NULL,
    SEMESTER            CHAR (10)       NOT NULL,
    YEAR                NUMBER          NOT NULL,
    FACULTY_ID          CHAR (8)        NOT NULL, DEFAULT 'No Body',
    CONSTRAINT SECTION_PK PRIMARY KEY (COURSE_ID, SECTION_
    NO, SEMESTER, YEAR),
    CONSTRAINT SECTION_FK1 FOREIGN KEY (COURSE_ID)
            REFERENCES COURSE_T (COURSE_ID)
            ON UPDATE CASCADE,
    CONSTRAINT SECTION_FK2 FOREIGN KEY (FACULTY_ID)
            REFERENCES FACULTY_T (FACULTY_ID)
            ON DELETE SET DEFAULT
            ON UPDATE CASCADE);
```

٧-١-١-٣-٣ قيود السجلات (Tuple Constraints):

بالإضافة إلـــى القيود التى من الممكن أن تفرض على الحقول والقيود التى تفرض لتأكيد السلامة المرجعية، توفر (SQL) قيود السجلات التى من الممكن أن تفرض على أكثر من حقل فى الجدول نفسه وفى الوقت نفسه. ولتعريف مثل هذه القيود تستخدم الكلمـــة (CHECK) فـــى نهاية تعريف الجدول. ويســـمى هذا النوع مـــن القيود بقيود الســـجلات؛ لأنه يفرض على كل ســـجل على حدة بشكل منفرد عند إضافة السجل أو التعديل عليه. فعلى سبيل المثال، لا يمكن أن تكون مادة دراسية معينة متطلباً دراسياً للمادة نفسها. ولتعريف هذا القيد نستخدم كلمة التحقق (CHECK) فى نهاية تعريفنا لجدول المواد الدراسية المتطلبة، كما يلى:

```
CREATE TABLE PREREQUISITE_T
        (COURSE_ID                  CHAR (7)       NOT NULL,
        PREREQUISITE_ID             CHAR (7)       NOT NULL,
    CONSTRAINT PREREQUISITE_PK PRIMARY KEY (COURSE_ID,
    PREREQUISITE_ID),
    CONSTRAINT PREREQUISITE_FK1 FOREIGN KEY (COURSE_ID)
                    REFERENCES COURSE_T (COURSE_ID),
    CONSTRAINT PREREQUISITE_FK2 FOREIGN KEY (PREREQUISITE_ID)
                    REFERENCES COURSE_T (COURSE_ID),
    CHECK (COURSE_ID <> PREREQUISITE_ID));
```

ويلاحظ أن القيد «لا يمكن أن تكون أية مادة دراسية متطلباً دراسياً للمادة نفسها» عبارة عن قيد يتعلق بأكثر من حقل ضمن الســـجل نفســـه. وعند إدخال سجل جديد لجدول المواد الدراسية المتطلبة أو تعديل أى سجل موجود فيه، يقوم نظام إدارة قاعدة البيانات بالتأكد من تحقق هذا القيد. وفى حالة خرق هذا القيد من قبل عملية تعديل أو إدخال فإن نظام إدارة قاعدة البيانات سيقوم برفض تنفيذ العملية.

٧-١-١-٣-٤ قيود عامة (Assertions):

القيود العامة هى قيود قد تتعلق بأكثر من جدول فى الوقت نفســـه. لذا فإن هذا النـــوع من القيود يجب أن ينطبق على أى حالـــة تكون فيها قاعدة البيانات عوضاً عن حالة الجدول كما هو الحال بالنســـبة لقيود الســـجلات، أو حالة الحقل كما هو الحال بالنســـبة لقيود الحقول. وتســـتخدم عبـــارة (CREATE ASSERTION) لتعريف القيود العامة. والشكل العام للقيد العام كما يلى:

```
CREATE ASSERTION Assertion_Name CHECK condition;
```

وعلى الرغم من أن القيود العامة تتطلب التعرف على القوة الحقيقية للاستفسارات فى لغة الاستفسار البنائية، إلا أننا نقدم المثال التالى، الذى سيتضح معناه أكثر عند استعراض الاستفسارات فى لغة الاستفسار البنائية. إن القيد العام الذى يهدف إليه المثال التالى هو التحقق من أن أى طالب فى الجامعة الأهلية لا يمكن أن يسجل (Registers) فى مواد دراسية يفوق عدد ساعاتها الإجمالية أكثر من اثنتى عشرة وحدة دراسية فى أى فصل كان. وللتأكد من تحقق هذا القيد على جميع حالات قاعدة البيانات فإن ذلك يتطلب التعرف على بيانات حقول تتبع لأكثر من جدول. ويقع على نظام إدارة قاعدة البيانات التأكد من أن هذا القيد حسب تعريفه التالى متحقق فى جميع حالات قاعدة البيانات.

```
CREATE ASSERTION REGISTRATION_CHECK
CHECK (Not Exists (
        (SELECT Sum (units)
        FROM Student_T s, Course_T c, Section_T t, Enrollment_T e
        Where s.Student_ID = e.Student_ID AND
        e.Course_ID = t.Course_ID AND
        e.Section_No = t.Section_No AND
        e.Semester = t.Semester AND
        e.Year = t.Year AND
        c.Course_ID = t.Course_ID
        Group By s.Student_ID, t.Semester) > 12));
```

وتجدر الإشارة هنا إلى بطء تنفيذ عمليات التعديل، سواء من خلال عمليات الإضافة، أو الحذف، أو التحديث على قاعدة البيانات عند استخدام القيود العامة، وذلك لضرورة مراجعة نظام إدارة قاعدة البيانات للقيود العامة فى كل مرة يتم التعديل على قاعدة البيانات.

٧-١-١-٣-٥ تعديل القيود والتحكم فى تطبيقها:

٧-١-١-٣-٥-١ تعديل القيود:

من الممكن إضافة، أو تعديل، أو إزالة القيود فى أى وقت كان. وتعتمد طريقة التعديل حسب القيد نفسه بمعنى إن كان متعلقاً بحقل، أو جدول، أو قاعدة البيانات. وكما أسلفنا سابقاً، إنه من الضرورى إعطاء القيود مسميات معينة حتى يمكن التعرف عليها ومن ثم تعديلها عند الرغبة فى ذلك.

٧-١-١-٣-٥-٢ إزالة القيود:

لإزالــة قيد ما من جدول تســتخدم عبارة إزالة قيــد (Drop Constraint) بالإضافة لعبارة تعديل جدول (Alter Table) كما هو موضح فى الشكل العام التالى للتعليمة:

```
ALTER TABLE Table_Name DROP CONSTRAINT Constraint_Name;
```

فعلى سبيل المثال، لو أردنا إزالة المفتاح الخارجى من جدول المجموعات الدراسية، والذى عرف على أســاس أنه قيد جدول، والمســمى (SECTION_FK2) الذى يشير إلى أعضاء هيئة التدريس المكلفين بتدريس مواد دراسية فى جدول أعضاء هيئة التدريس كما هو مبين فى تعريف جدول المجموعات الدراسية التالى:

```
CREATE TABLE SECTION_T
  (COURSE_ID     CHAR (7)      NOT NULL,
  SECTION_NO     NUMBER        NOT NULL,
  SEMESTER       CHAR (10)     NOT NULL, CONSTRAINT SEM_NAME
                    CHECK (VALUE IN 'FALL', 'SPRING', 'SUMMER'),
  YEAR           NUMBER        NOT NULL,
  FACULTY_ID     CHAR (8)      NOT NULL,
  CONSTRAINT SECTION_PK PRIMARY KEY (COURSE_ID, SECTION_NO,
                        SEMESTER, YEAR),
  CONSTRAINT SECTION_FK1 FOREIGN KEY (COURSE_ID)
                        REFERENCES COURSE_T (COURSE_ID),
  CONSTRAINT SECTION_FK2 FOREIGN KEY (FACULTY_ID)
                        REFERENCES FACULTY_T (FACULTY_ID));
```

فإنه يمكن استخدام عبارة إزالة قيد السلامة المرجعية المذكور أعلاه كما يلى:

```
ALTER TABLE SECTION_T DROP CONSTRAINT SECTION_FK2;
```

كما يمكن إزالة القيد المفروض على حقل الفصل الدراسى (Semester)، الذى عرف على أســاس أنه قيد حقل، بحيث يجب أن تكون قيمتــه إما الخريف (Fall)، أو الربيع (Spring)، أو الصيف (Summer)، كما يلى:

```
ALTER TABLE SECTION_T DROP CONSTRAINT SEM_NAME;
```

٧-١-١-٣-٥-٣ إضافة القيود:

لإضافة القيود تستخدم عبارة إضافة قيد (ADD CONSTRAINT) عوضاً عن عبارة إزالة قيد. ولو أردنا إعادة تعريف قيد السلامة المرجعية الذى تم إلغاؤه، فإنه يمكن إعادة تعريفه من جديد كما يلى:

```
ALTER TABLE SECTION_T ADD CONSTRAINT SECTION_FK2
FOREIGN KEY (FACULTY_ID) REFERENCES FACULTY_T (FACULTY_ID);
```

كما يمكن إعادة تعريف القيد المفروض على حقل الفصل الدراسى بعد إزالته بحيث يمكن أن تكون قيمته واحدة من أربع قيم عوضاً عن ثلاث، وذلك من خلال إضافة فصل الشتاء (Winter) الذى من الممكن أن تنفذ فيه الجامعة بعضاً من موادها الدراسية، كما يلى:

```
ALTER TABLE SECTION_T ADD CONSTRAINT SEM_NAME
    CHECK (SEMESTER IN ('FALL', 'SPRING', 'SUMMER', 'WINTER'));
```

ويلاحظ فى تعريفنا للقيد السابق أنه قد أصبح قيد جدول عوضاً عن قيد حقل، وذلك لكون لغة الاستفسار البنائية لا توفر طريقة لتعريف قيود حقول بعد إنشاء الجداول.

٧-١-١-٣-٥-٤ تعطيل عمل القيود واستعادة العمل بها:

يمكن تعطيل العمل بأىٍّ من القيود مع الاحتفاظ بها لإعادة العمل بها واستخدامها لاحقاً. ولتعطيل العمل بقيد ما، تستخدم عبارة تعطيل القيد(Disable Constraint). أما عبارة إعادة العمل بالقيد (Enable Constraint) فتستخدم لإعادة العمل بالقيد. وتختلف هاتان العمليتان عن عمليتى حذف القيود وإعادة تعريفها لكون هاتين العمليتين لا تلغيان قيوداً معرفة أو تقوم بتعريف قيود جديدة وإنما تستخدم لإيقاف العمل بالقيود لفترة ما، ومن ثم إعادة العمل بها مرة أخرى. ويمكن تعطيل عمل قيد السلامة المرجعية المتعلق بأعضاء هيئة التدريس فى جدول المجموعات الدراسية كما يلى:

```
ALTER TABLE SECTION_T DISABLE CONSTRAINT SECTION_FK2;
```

ويمكن استعادة العمل بالقيد كما يلى:

```
ALTER TABLE SECTION_T ENABLE CONSTRAINT SECTION_FK2;
```

ويستفاد من العمليتين السابقتين بشكل خاص عند وجود سلسلة من قيود السلامة المرجعية بين جداول قاعدة البيانات مما يتطلب إدخال البيانات فى جداول قاعدة البيانات وفق ترتيب معين، وإلا فشلت عملية إدخال البيانات، أو فى حالة وجود حلقة بين قيود السلامة المرجعية لأكثر من جدول يستعصى فى ظل وجودها إدخال البيانات للجداول التى فرضت عليها هذه الحلقة من قيود السلامة المرجعية. فى مثل هذه الحالة، يمكن تعطيل العمل بقيود السلامة المرجعية لحين إدخال البيانات فى الجداول التى تستلزم ترتيباً معيناً فى إدخال بياناتها أو تكون مرتبطة بحلقة من القيود. وبعد إدخال البيانات فى الجداول يتم إعادة العمل بالقيود من جديد. ويتجلى أكثر استخدامات العمليتين السابقتين أهمية عند نقل كميات كبيرة من البيانات من قاعدة بيانات إلى قاعدة بيانات أخرى.

٧-١-١-٣-٥-٥ تأخير العمل بالقيود (Deferring the Checking of Constraints):

فى الكثير من الأحيان تظهر حاجة إلى تعطيل العمل بالقيود. وتظهر هذه الحاجة إما لتحسين أداء النظام أو حتى نتمكن من التغلب على ما يعرف بالقيود المرتبطة بشكل حلقى. فعل سبيل المثال، يمكن أن يستدعى إضافة سجل فى جدول ما وجود سجل فى جدول آخر ويستدعى السجل الثانى وجود سجل فى جدول ثالث، إلى أن نصل إلى حالة يستدعى فيها السجل الأخير وجود سجل فى الجدول الذى نحن بصدد إضافة سجل إليه. وهذه الحلقة من القيود لا يمكن التغلب عليها إلا من خلال تعطيل العمل ببعض القيود، وبعد إضافة السجل (أو السجلات المطلوبة)، يعاد العمل بها من جديد كما أسلفنا أعلاه. والطريقة الأخرى هى تأخير العمل بالقيود لحين الانتهاء من التعامل مع قاعدة البيانات. وهذه الطريقة هى المتبعة عادة عندما نتعامل مع قاعدة البيانات من خلال ما يعرف بالمعاملات (Transactions). والمعاملة هى برنامج حاسوبى يتعامل فى بعض أجزائه مع قاعدة البيانات. ويمكن أن تنتهى المعاملة بإحدى طريقتين: إما أن تنتهى المعاملة بشكل كامل وتنعكس نتائجها كافة على قاعدة البيانات، أو تفشل المعاملة فى أثناء عملية تنفيذها لسبب ما، ولا ينعكس أى من نتائجها على قاعدة البيانات. وإذا ما أخذنا بهذا التعريف للمعاملات، فإن لغة الاستفسار البنائية توفر طريقة لتأخير العمل بالقيود لحين انتهاء المعاملات.

فعلى سبيل المثال، يمكن تعريف قيود السلامة المرجعية فى جدول المجموعات كما يلى:

```
CREATE TABLE SECTION_T
   (COURSE_ID         CHAR (7)      NOT NULL,
   SECTION_NO         NUMBER        NOT NULL,
   SEMESTER           CHAR (10)     NOT NULL,
   YEAR               NUMBER        NOT NULL,
   FACULTY_ID         CHAR (8)      NOT NULL,
   PRIMARY KEY (COURSE_ID, SECTION_NO, SEMESTER, YEAR),
   CONSTRAINT SECTION_FK1 FOREIGN KEY (COURSE_ID)
                              REFERENCES COURSE_T
                              DEFERRABLE INITIALLY DEFERRED,
   CONSTRAINT SECTION_FK2 FOREIGN KEY (FACULTY_ID)
                              REFERENCES FACULTY_T
                              DEFERRABLE INITIALLY IMMEDIATE);
```

ويعنى قيد السـلامة المرجعيـة الأول والمسـمى (SECTION_FK1) أن قيد المفتاح الخارجـى قد تم تعريفه على أنه قابل للتأخير من حيث التأكد من سـلامته المرجعية لحين انتهاء المعاملة التى قد تتعامل معه، وأنه مبدئياً مؤخر العمل به. أما القيد الثانى والمسـمى (SECTION_FK2) فقد تم تعريفه على أنه قابل للتأخير من حيث التأكد من سـلامته المرجعية لحين الانتهاء من المعاملة التى قد تتعامل معه، ولكنه سيطبق (ويتم التأكد من السـلامة المرجعية) عند إضافة أى سـجل جديد للجدول أو تحديث لحقل المفتاح الخارجى لسـجل موجود أو تعديل على المفتاح الرئيسى (من خلال الحذف أو التحديث) فى جدول أعضاء هيئة التدريس ما لم يطلب تأخيره من خلال المعاملة التى تتعامل معه. وسنتطرق لمفهوم المعاملات (Transactions) فى نظم قواعد البيانات بشىء من التفصيل فى الجزء (٩-١).

ويمكـن تأخير العمل بالقيد الثانى من خلال المعاملة التى تتعامل معه باسـتخدام التعليمة التالية:

```
SET CONSTRAINT SECTION_FK2 DEFERRABLE;
```

وعنـد تأخير العمل بالقيد، يتم التحقق من سـلامته المرجعية عند انتهاء المعاملة عوضـاً عـن التحقق منه فى أثناء تعامل المعاملة مع سـجلات الجدول عند إجراء كل تعديل أو إضافة إلى الجدول. كما يمكن أن يتم تغيير العمل بالقيد الأول بحيث يصبح غير مؤخر من خلال المعاملة كما يلى:

```
SET CONSTRAINT SECTION_FK1 IMMEDIATE;
```

وفى هذه الحالة، يتم التحقق من السـلامة المرجعية بعد كل تعديل على سـجلات الجدول أو الإضافة إليه عوضاً عن التحقق من السلامة المرجعية فى نهاية المعاملة.

٧-١-١-٤ إنشاء منظور (Create View):

تمكن لغة الاستفسـار البنائية من تعريف المنظـورات. والمنظور عبارة عن جدول تخيلى (VIRTUAL TABLE) له تعريف ضمن هيكل قاعدة البيانات، ولكنه لا يحتوى على بيانات مخزنة فيه بشكل دائم، وإنما يستمد بياناته من الجداول (الأساسية) الموجودة فى قاعدة البيانات. ويقوم نظام إدارة قاعدة البيانات بتعبئة البيانات المناسبة للمنظور بمجرد التعامل معه (من خلال إجراء عملية اختيار أو إضافة أو تحديث عليه) من أحد المستفيدين المخولين بالتعامل معه. وبمجرد الانتهاء من التعامل مع المنظور، يقوم نظام إدارة قاعدة البيانات بإتلاف ما يحتويه من بيانات (بعد إجراء التعديلات المطلوبة، إن وجدت، على الجدول أو الجداول الأساسية الذى استمد المنظور بياناته منها). وعند التعامل مع المنظور مرة أخرى، تتم تعبئته بالبيانات المناسـبة مرة أخرى، وهكذا. إلا أن بعض نظم قواعد البيانات تقوم بالمحافظة على بيانات المنظور لبعض الوقت عوضاً عن إتلافها بمجرد انتهاء مسـتخدم ما من الانتهاء من التعامل مع المنظور، وذلك على أمل أن يأتى مسـتفيد آخر يرغب فى التعامل مع بيانات المنظور. وتسـمى الطريقة الثانيـة بالمنظورات المخزنة (Materialized Views). وتعد هذه الطريقة مفيدة فى بعض الحـالات لأنها تعفى من تعبئة البيانات للمنظور كلما اسـتدعت الحاجة الرجوع إليه، وخاصة أن تكلفة تعبئة بيانات المنظور قد تكون كبيرة جداً من حيث الوقت اللازم من الحاسب الآلى لتعبئتها.

وتتمثل أهمية المنظورات فى شكلين: الأول منهما عندما يكون هنالك عمليات اختيار معقدة قد تكون مصدراً للأخطاء عن محاولة كتابتها بشكل متكرر من قبل المستفيدين أو فـى برامج التطبيقات، والثانى منهما يتمثل فى كون المنظورات توفر حماية للبيانات من الاسـتخدامات التى لا يُرغَب فى التصريح بها لبعض المسـتفيدين. فعلى سـبيل المثـال، يمكن كتابة منظور يحتوى علـى المعلومات المالية لأعضـاء هيئة التدريس فى الجامعة الأهلية، ومنظور آخر لا يحتوى على مثل هذه المعلومات بحيث تعطى صلاحية التعامل مع المنظور الأول للمستفيدين فى إدارة الشئون المالية وبرامج التطبيقات المالية، فـى حين تعطى صلاحيات التعامل مـع المنظور الثانى للمسـتفيدين فى إدارة القبول والتسـجيل ورؤساء الأقسام العلمية، سـواء بشكل مباشر أو من خلال التطبيقات التى

يستخدمونها للتعامل مع قاعدة البيانات. ولتعريف مثل هذين المنظورين يمكن استخدام تعليمة الاختيار (التى سيتم شرحها بالتفصيل فى الجزء ٧-٢-١) كما يلى:

```
CREATE VIEW FACULTY_FINANTIAL_INF_V AS
SELECT FACULTY_ID, FNAME, LNAME, SALARY, DEPARTMENT_ID
FROM FACULTY_T;
```

وعند استعراض محتويات المنظور السابق باستخدام تعليمة الاختيار على المنظور، كما يلى:

```
SELECT *
FROM FACULTY_FINANTIAL_INF_V;
```

تكون النتيجة جميع الحقول التى تهم المتعاملين مع المنظور فى إدارة الشئون المالية، كما يلى:

```
FACULTY_ FNAME        LNAME            SALARY DEPART
-------- ------------ ------------ ---------- ------
200      Khalid       Aloufi            35000 MATH
220      Fahad        Alhamid           25900 MATH
310      Saleh        Aleesa            30000 CS
320      Mohammed     Alhamad           44000 CS
330      Ghanim       Alghanim          44500 CS
340      Ibraheem     Alsaleh           25000 CS
400      Ahmad        Alotaibi          33900 CHEM
420      Saleh        Alghamdi          44600 CHEM
500      Yahya        Khorshid          36700 ENGL
540      Salem        Alhamad           40000 ENGL
560      Salman       Albassam          33800 ENGL
600      Turki        Alturki           27800 STAT
640      Fahad        Alzaid            44300 STAT
660      Saud         Alkhalifa         44900 STAT
710      Mahmood      Alsalem           31900 PHYS
730      Mishal       Almazid           29800 PHYS
770      Sultan       Aljasir           43300 PHYS
800      Ali          Albader           45300 EE
810      Saad         Alzhrani          44200 EE
850      Ahmad        Alsabti           33900 EE
```

أما المنظور الثانى الذى لا يحتوى على المعلومات المالية فيمكن تعريفه كما يلى:

```
CREATE VIEW FACULTY_V AS
SELECT FACULTY_ID, FNAME, LNAME, DOB, PHONE_NO, DEPARTMENT_ID
FROM FACULTY_T;
```

ويمكن استعراض محتويات المنظور السابق باستخدام تعليمة الاختيار على المنظور، كما يلى:

```
SELECT *
FROM FACULTY_V;
```

وتكون النتيجــة جميع الحقول التى تهم المتعاملين مع المنظور من غير العاملين فى إدارة الشئون المالية، كما يلى:

FACULTY_	FNAME	LNAME	DOB	PHONE_NO	DEPART
200	Khalid	Aloufi	22-MAY-63	454-2341	MATH
220	Fahad	Alhamid	07-OCT-70	456-7733	MATH
310	Saleh	Aleesa	13-SEP-66	454-8932	CS
320	Mohammed	Alhamad	13-MAY-65	454-5412	CS
330	Ghanim	Alghanim	12-AUG-69	456-2234	CS
340	Ibraheem	Alsaleh	20-JAN-70	454-1234	CS
400	Ahmad	Alotaibi	17-MAY-71	454-4563	CHEM
420	Saleh	Alghamdi	13-FEB-69	454-2233	CHEM
500	Yahya	Khorshid	12-MAR-65	456-2221	ENGL
540	Salem	Alhamad	11-SEP-72	456-3304	ENGL
560	Salman	Albassam	13-SEP-68	454-7865	ENGL
600	Turki	Alturki	23-JUL-75	456-7891	STAT
640	Fahad	Alzaid	12-MAY-71	456-3322	STAT
660	Saud	Alkhalifa	13-AUG-72	454-9856	STAT
710	Mahmood	Alsalem	19-FEB-73	456-3323	PHYS
730	Mishal	Almazid	17-SEP-75	454-2343	PHYS
770	Sultan	Aljasir	13-MAY-70	456-3212	PHYS
800	Ali	Albader	22-JUN-66	456-7812	EE
810	Saad	Alzhrani	17-OCT-67	454-5578	EE
850	Ahmad	Alsabti	15-APR-73	456-0120	EE

ويجســد المنظور فى لغة الاستفسار البنائية مفهوم عدم الاعتمادية المنطقية، الذى تم التطــرق إليه فى الفصل الأول، بين المنظورات الخارجية (External Views) والمنظور المفاهيمــى (أو المنطقى) لقاعدة البيانات، بحيث إن أى تغيير للجداول المكونة لقاعدة البيانات لا يؤثر فى نظرة المستفيدين لقاعدة البيانات. فعلى سبيل المثال، عند تجزئة جدول إلى جدولين أو أكثر أو إضافة حقول جديدة للجدول فإن المستفيدين، باستخدام المنظــورات، قد لا يلحظون مثل هذا التغيير فى هيكل قاعدة البيانات. لهذا الســبب فإن إداريى قواعد البيانات العلاقية كثيراً ما يقرنون بين كل جدول من جداول قاعدة

البيانات بمنظور له. ويتم اســتخدم المنظور المصاحب لجدول ما من خلال التعليمات نفســها التى تســتخدم مع الجداول دون أى تفريق كما لو أنه يتم اســتخدام الجدول الأساســى، وليس المنظور المصاحب له. ويعنى هذا أنه يتم اســتخدام المنظور من قبل المســتفيدين ومن قبــل التطبيقات المبنية على قاعدة البيانــات عوضاً عن التعامل مع الجدول الأساسى بشكل مباشر. وبهذه الطريقة يمكن إجراء أى تعديلات قد تتطلبها مراحل مســتقبلية على جداول قاعدة البيانات دون الحاجة لإجراء تعديلات مصاحبة علــى نظم التطبيقات أو التعليمات التى تعوَّد المســتفيدون علــى تنفيذها على قاعدة البيانات. وباســتخدام المنظورات يكتفى بتعديل تعريــف المنظور الذى جرى التعارف على استخدامه من قبل المستفيدين وبرامج التطبيقات (من خلال إزالته ومن ثم إعادة تعريفه من جديد تحت المسمى نفسه) دون الحاجة إلى التعديل على برامج التطبيقات بحيث تنعكس هذه التعديلات التى أجريت على الجداول الأساســية لقاعدة البيانات علــى برامج التطبيقات مما يعنى وجود اســتقلالية (أو عــدم اعتمادية) منطقية بين برامج التطبيقات والتصميم المنطقى لقاعدة البيانات.

وتجدر الإشارة إلى أنه لا يقتصر تعريف المنظور بحيث يقترن بجدول واحد فقط من جــداول قاعدة البيانات، وإنما يمكن تعريفه بحيث يقتــرن بأكثر من جدول واحد. كما أنه من الممكن أن تســتخدم دوال التجميع (التى ســنتطرق إليها لاحقاً) فى تعريف بعض حقول المنظور. فعلى سبيل المثال، يمكن تعريف المنظور التالى الذى يقترن بأعضاء هيئة التدريس الذين يعملون فى قسم الحاسب الآلى (فى جدول أعضاء هيئة التدريس) والمواد الدراسية المؤهل لتدريسها هؤلاء الأعضاء (فى جدول المؤهلات التدريسية)، كما يلى:

```
CREATE VIEW CS_FACULTY_QUALIFICATION_V AS
SELECT FACULTY_T.FACULTY_ID, FNAME, LNAME, COURSE_ID,
DATE_QUALIFIED
FROM FACULTY_T, QUALIFICATION_T
WHERE FACULTY_T. FACULTY_ID = QUALIFICATION_T. FACULTY_ID AND
      FACULTY_T. DEPARTMENT_ID = 'CS';
```

وتعرف التعليمة السابقة منظوراً باسم «المؤهلات التدريسية لأعضاء هيئة التدريس الذين يعملون فى قسم الحاسب الآلى»، التى يلاحظ فيها استخدام الحرف "V" للدلالة على تعريف منظور عوضاً عن جدول، بحيث يقترن المنظور بجدولين كما أسلفنا أعلاه. وعند الاستفسار عن المحتويات التى يقترن بها هذا المنظور فى الجداول الأساسية من خلال التعليمة التالية:

```
SELECT *
FROM CS_FACULTY_QUALIFICATION_V;
```

تكون نتيجة التعليمة السابقة التى تسترجع محتويات المنظور حسب تعريفه أعلاه، كما يلى:

```
FACULTY_ FNAME       LNAME        COURSE_ DATE_QUAL
-------- ----------- ------------ ------- ---------
310      Saleh       Aleesa       CS101   05-JUN-95
320      Mohammed    Alhamad      CS102   09-AUG-95
320      Mohammed    Alhamad      CS103   03-AUG-96
330      Ghanim      Alghanim     CS104   02-SEP-97
340      Ibraheem    Alsaleh      CS105   02-DEC-97
```

ولأنــه من الممكن تعريــف المنظورات بحيث تقترن بأكثر مــن جدول، كما يمكن أن تحتــوى تعاريفها على دوال تجميــع، فإنه قد يتعذر إجراء عمليات التعديل (من حذف أو إضافة أو تحديث) على بياناتها بمعنى عدم إمكانية تعديل بيانات الجداول المقترنة بالمنظــورات. والقاعــدة العامة التى تمكــن من إجراء عمليات التعديــل على بيانات المنظــورات هى: يمكــن إجراء عمليات التعديل على بيانـــات منظور مادام للقيم التى أجــرى التعديل عليها فى المنظور ما يكافئها مــن حقول فى الجداول المقترنة بتعريف المنظور دون أى التباس بين هذه الحقول. وبناءً على هذه القاعدة، يتعذر بشــكل عام إجراء التعديل على أى منظور يكون من ضمن حقوله حقولاً يستخدم فى تعريفها دوال تجميع. والسبب وراء ذلك هو عدم اقتران الحقول الممثلة لدوال التجميع، ضمن حقول المنظــور، بأى من حقول الجداول التى يقترن بها المنظور. وفى مثل هذه الحالة يمكن إجراء عمليات الاختيار (أو الاسترجاع) فقط على مثل هذه المنظورات.

٧-١-١-٥ إنشاء فهرس (Create Index):

الفهرس، بشكل عام، عبارة عن طريقة تمكن من الوصول إلى الشىء المطلوب بسرعة كبيرة. وتســتخدم الفهرسة بشــكل مكثف فى حياتنا اليومية حيث نجدها مستخدمة، على ســبيل المثال، فى المكتبات، والمســتوصفات (أو المستشفيات) الطبية، إلخ. وبدون الفهارس يضطر الشــخص الذى يبحث عن كتاب فى مكتبة ما، على ســبيل المثال، إلى الســير فى ممرات المكتبة كافة بشــكل متسلســل وقراءة عنوان كل كتاب فى كل ممر وبشــكل متسلســل أيضاً حتى يجد الكتاب الذى يبحث عنه. وفى حالة كان الشخص يبحث عن كتاب ليس موجوداً أصلاً فى المكتبة فإنه ســيضطر إلى المرور بكافة الكتب

الموجودة حتى يتيقن من عدم وجود الكتاب. على النقيض من ذلك فإن الفهارس تعجل فــى عملية البحث وبشــكل كبير عن الشــىء المطلوب. وتقتضــى الفهارس (١) وجود تصنيف معين للأشــياء الموجودة وترتيب الأشــياء وفقاً للتصنيف، و(٢) ترتيب الأشياء وفق نمط معين على أرض الواقع. فعلى ســبيل المثال، قد يتم ترتيب الكتب فى المكتبة وفــق المقياس الدولى لترقيم الكتب ((International Standard Book Number (isbn))، ويبنى على هــذا الترتيب فهارس مختلفة من ضمنها فهــرس الموضوعات، وفهرس المؤلفين، وفهرس دور النشــر، إلخ. ومعنى هذا أننا قد قمنا بترتيب الكتب فى المكتبة وفق نمط معين على أرض الواقع، وأنشــأنا فهارس للتصنيفات المناســبة. فى هذه الحالة يمكن البحــث عــن أى كتاب وفق أحد المعايير التى صنفت عليهــا. فمثلاً، يمكن البحث عن الكتــب التى ألفت من قبل مؤلف ما فى فهرس المؤلفين. وحيث إن هذا الفهرس مرتب أبجدياً حســب أسماء المؤلفين، فإن عملية البحث ستتم بشكل أسرع لأن عملية البحث لن تتم بالضرورة بشكل متسلسل فى الفهرس. وعند العثور على الكتاب ضمن فهرس المؤلفــين، يتم التعرف على رقمه. بعد ذلك يتم الذهاب للممر الذى يحتوى على مكان الكتب الذى يتضمن رقم الكتاب المطلوب والســير فى ذلك الممر بشــكل متسلسل حتى العثــور على الكتاب. أمــا فى حالة البحث عن كتاب لمؤلف ما والكتاب غير متوافر فى المكتبــة، فإنه يكتفى بالبحث فى فهرس المؤلفين. وعند عدم العثور على اســم المؤلف ضمن أسماء المؤلفين، يتم التوقف عن البحث لعدم توافر الكتاب فى المكتبة.

والفهــرس فى نظم قواعد البيانات ما هو إلا هيــكل بيانات (Data Structure) يبنى علــى حقل أو أكثر من حقول الجدول تمثل فكرته الرئيســية ما نجده من فهارس فى حياتنــا اليومية. وتســاعد الفهارس فى الوصول إلى الســجل المطلوب فى جدول ما خاصة عندما يتم بناؤه على حقل معين وتكون عمليات الاستفسار تتضمن مقارنة بين الحقل الذى بنى عليه الفهرس وقيمة ثابتة مثل «رقم السجل المدنى = ١٢٣٤٥٦٧٨٩»، بحيث إن رقم الســجل المدنى هو الحقل الذى بنى عليه الفهرس فى جدول الموظفين، على سبيل المثال.

وعلى الرغم من أن تعليمة إنشاء الفهارس لم تعد من ضمن لغة الاستفسار البنائية لكونها لا تتعلق بتعريف هياكل قاعدة بيانات أو تداول محتوياتها وإنما وســيلة لتسريع وتحســين أداء تظم إدارة قواعــد البيانات، إلا أن غالبية نظــم إدارة قواعد البيانات العلاقية تحتوى على تعليمة لإنشــاء الفهارس. وتستخدم عبارة إنشاء فهرس (Create Index) حسب الشكل العام التالى للتعليمة:

```
CREATE [UNIQUE] INDEX IndexName ON TableName
(ColumnName [order][,ColumnName [order]] .... );
```

- تستخدم الكلمة الاختيارية فريد [UNIQUE] عند الرغبة فى إنشاء فهرس يكون الحقل الذى أنشئ عليه الفهرس حقلاً فريداً لا تتكرر قيمه فى سجلات الجدول.
- يمكن تحديد أكثر من عمود لإنشاء فهرس مركب.
- يقصد بالترتيب [order] الطريقة التى سيتم فيها ترتيب الفهرس، فإما أن يكون تصاعدياً (ASC) أو تنازلياً (DESC) بحيث يكون الترتيب الافتراضى هو الترتيب التصاعدى.
- يمكن إنشاء أى عدد من الفهارس للجدول الواحد، سواء كانت مبنية على حقول فريدة (UNIQUE) أو حقول يسمح فيها بالقيم المتكررة (Duplicates).

ولإنشاء فهرس فريد باسم فهرس الطلبة (Students_IDX) على الرقم الدراسى للطلبة فى جدول الطلبة بشكل تصاعدى تستخدم تعليمة إنشاء فهرس كما يلى:

```
CREATE UNIQUE INDEX STUDENTS_IDX
ON STUDENT_T (STUDENT_ID );
```

أما إذا أردنا إنشاء فهرس بأسماء الطلبة تحت مسمى (Student_names_IDX) على اسم العائلة للطلبة بشكل تصاعدى، والاسم الأول لهم بشكل تنازلى فتستخدم التعليمة التالية:

```
CREATE INDEX STUDENT_NAMES_IDX
ON STUDENT_T (LName ASC, FName DESC);
```

٧-١-٢ تعليمة الإزالة (DROP STATEMENT):

تستخدم تعليمة الإزالة (DROP) لحذف العناصر ذات المسميات من هيكل قاعدة البيانات مثل الجداول، ومدى القيم، والقيود، والمنظورات، والفهارس ... إلخ. كما أنه يوجد نوعان من الخيارات السلوكية لتعليمة الإزالة وهى «التتابع» (CASCADE)، و«التقييد» (RESTRICT). فإذا ما أردنا حذف (تعريف) قاعدة بيانات بأكملها بما فى ذلك ما تحتويه من جداول وقيود وجميع العناصر الأخرى المكونة لقاعدة البيانات يمكن استخدام تعليمة الإزالة مصحوبة بخيار التتابع كما يلى:

```
DROP SCHEMA Company CASCADE;
```

أما إذا تم اســتخدام خيار التقييد (Restrict)، الذى يمثل الحالة الافتراضية، ضمن تعليمة الإزالة فإنه يتم إزالة (تعريف) قاعدة البيانات فقط، وذلك عند عدم وجود أى عنصر فيها. أما إذا وجد فيها عنصر أو أكثر فلن تنفذ عملية الإزالة.

كذلك هو الحال عند اســتخدام التعليمة لإزالة جدول ما حيث يتم استخدام خيار التتابــع لإزالة محتويات الجدول، وتعريفه، والقيود المفروضة عليه، والمنظورات المبنية عليه. أما إذا تم اســتخدام خيار التقييد، فإن عملية الإزالة لا تتم إلا فى حالة عدم وجود ما يرتبط بالجدول من عناصر أخرى ضمن قاعدة البيانات. كما يمكن استخدام تعليمة الإزالة لحذف أى من مكونات قاعدة البيانات الأخرى مثل المنظورات، والقيود، والفهارس، وقيود المدى التى يتم تعريفها. وكما أســلفنا أعلاه فإن الحالة الافتراضية لعمليــة الإزالة هى التقييد، بمعنى عدم حذف العنصــر إذا احتوى على أية بيانات أو ارتبط بأى من العناصر الأخرى لقاعدة البيانات. ويمثل الشكل التالى ثلاث تعليمات: الأولى تمثل عملية إزالة جدول المواد الدراســية (Course_T)، على افتراض عدم وجود أى ســجلات فى الجدول، والثانية لإزالة فهرس بمسمى (Course_IDX)، والثالثة لإزالة منظور بمسمى (Course_V).

```
DROP TABLE COURSE_T;
DROP INDEX COURSE_IDX;
DROP VIEW COURSE_V;
```

٧-١-٣ تعليمة التعديل (ALTER STATEMENT):

إن تعريف أى جدول أو عنصر ذى مسمى ضمن قاعدة البيانات يمكن التعديل عليه باســتخدام تعليمة التعديــل (ALTER). ومن التعديلات التــى بالإمكان إجراؤها على جدول ما إضافة حقل جديد، أو حذف حقل من حقول الجدول، أو تغيير تعريف حقل مــا ضمن الجدول، أو إضافة أو حذف قيد. فعلى ســبيل المثــال، يمكن إضافة حقل جديد لجدول أعضاء هيئة التدريس يعكس تاريخ الحصول على آخر درجة علمية تكون بياناته من نوع تاريخ، كما يلى:

```
ALTER TABLE FACULTY_T ADD Graduation_Date DATE;
```

وبعـد تعريف الحقـل الجديد، يجب إدخال بياناته إما من خلال اسـتخدام عبارة القيمـة الافتراضية (DEFAULT) أو من خلال اسـتخدام تعليمة التحديث (UPDATE) على البيانات التى سنتطرق إليها لاحقاً. أما إذا لم يتم تعريف قيمة افتراضية للحقل الجديد فسـتكون قيمته لجميع السـجلات فى الجدول غير معرفة. ويعنى هذا عدم إمكانية فرض القيد «غير المعرف» (NOT NULL) على الحقل.

ولإزالـة حقـل ما من جدول، فإنه يجب تعريف سـلوك عملية الإزالـة: إما تتابع (CASCADE) وإما تقييد (RESTRICT). وعند استخدام تتابع، تتم إزالة الحقل والقيود المعرفة عليه كافة، والمنظورات التى تسـتخدمه فى تعريفها، وذلك بشـكل تلقائى فى أثناء إزالة الحقل. أما إذا تم استخدام خيار التقييد، فتكون عملية الإزالة ناجحة فقط عندما لا يكون هنالك أى عنصر من عناصر قاعدة البيانات مستخدماً لهذا الحقل فى تعريفـه أو الرجوع إليه. وتمثل التعليمة التالية طريقة إزالة حقل تاريخ الميلاد (DOB) من جدول أعضاء هيئة التدريس:

```
ALTER TABLE FACULTY_T DROP DOB;
```

كمـا أن تعليمـة التعديل تمكن أيضاً من حذف القيمـة الافتراضية لحقل ما، كما يوضـح المثال التالى الذى يقوم بإزالة القيمة الافتراضية من حقل رقم عضو التدريس المعرف فى جدول المجموعات الدراسية:

```
ALTER TABLE SECTION_T ALTER FACULTY_ID DROP DEFAULT;
```

أمـا تعليمة التعديل التالية فتوضح طريقة تعريف قيمة افتراضية لحقل رقم عضو هيئـة التدريس فى جـدول المجموعات الدراسـية، بحيث تكون القيمـة الافتراضية 'AAAAAAAA':

```
ALTER TABLE SECTION_T ALTER FACULTY_ID SET DEFAULT 'AAAAAAAA';
```

ويمكن اسـتخدام تعليمة التعديل لتغيير القيود من حيـث حذفها أو تعريفها، كما سبق أن أوضحنا فى الجزء المتعلق بتوصيف القيود وأنواعها.

ويحتوى الملحق رقم (١) فى جزئه السـادس (ملحق رقم (١) - ٦) على بناء لجميع جداول قاعدة بيانات الجامعة الأهلية والبيانات التى تحتويها القاعدة فى بيئة أوراكل. وتمثـل قاعدة البيانات هذه محور الغالبية العظمى من التمارين التطبيقية الواردة فى هذا الفصل وفى الفصل الثامن.

٧-٢ لغة معالجة البيانات ((Data Manipulation Language (DML)):

٧-٢-١ تعليمة الاختيار (SELECT STATEMENT):

تعـد تعليمة الاختيار (SELECT) واحدة من أهم تعليمات لغة الاستفســار البنائية. وتستخدم تعليمة الاختيار لاسترجاع البيانات الموجودة فى قاعدة البيانات. كما تجدر ملاحظــة أن تعليمة الاختيار فى لغة الاستفســار البنائية ليســت ذات أية علاقة مع عملية الاختيار فى الجبر العلاقى الذى ســبق التطرق إليــه فى الفصل الثالث. كما تجدر أيضاً ملاحظة وجود فرق جوهرى بين لغة الاستفسار البنائية والنموذج العلاقى الرســمى. ويكمن هذا الفرق فى أن لغة الاستفسار البنائية تتعامل مع الجداول على أساس أنها حقائب (Bags) (أو مجموعات متكررة (Multiset)) عوضاً عن مجموعات من السجلات كما هو الحال فى النموذج العلاقى الرسمى. ويعنى ذلك أنه من الممكن أن تتكرر الســجلات فى الجداول باستخدام لغة الاستفسار البنائية، الأمر الذى لا يمكن فى النموذج العلاقى الرســمى. ومع هذا يمكــن أن تقيد الجداول بحيث لا يمكن أن تتكرر السجلات فيها، وذلك باستخدام المفتاح الرئيسى الذى لا يمكن أن تتكرر قيمه، كما يمكن أن تقيد القيم المســترجعة من جدول ما باســتخدام تعليمة الاختيار، بحيث لا تتكرر ضمن نتيجة العملية، وذلك باســتخدام عبارة (DISTINCT) التى تقوم بحذف السجلات المتكررة من نتيجة العملية.

ويوجد لتعليمة الاختيار العديد من الأشــكال التى قد يكون بعضها معقداً بشــكل كبير، إلا أننا ســنبدأ بأبسط أشــكال التعليمة وسنواصل شــرح التعليمة وصولاً إلى الأشكال المعقدة منها. أما الشكل العام للتعليمة فهو كما يلى:

```
SELECT  [ DISTINCT ] ColumnName(s)
FROM  Table(s)
[ WHERE  Condition ]
[ GROUP  BY ColumnName(s) ]
[ HAVING Condition ]
[ ORDER  BY ColumnName(s) ] ;
```

وتدل العبارات داخل الأقواس المربعة، فى الشــكل العام للتعليمة، على أنها عبارات اختيارية يتم استخدامها حسب الحاجة، مما يعنى أنه لا يجب استخدامها فى أشكال التعليمة كافة. إلا أنه عند اســتخدام العبــارات الاختيارية فإنه يجب إدراجها ضمن التعليمة بالترتيب نفسه الوارد فى الشكل العام للتعليمة المذكور أعلاه.

٧-٢-١-١ اختيار أعمدة محددة من جدول:

إن أبسط شكل لتعليمة الاختيار (SELECT) يكون عند استخدامها لاختيار حقل أو أكثر من جدول معين. وعند استخدام التعليمة فى هذه الصورة فإنه يعنى إظهار قيم الحقول التـــى تم تحديدها ضمن عبارة الاختيـــار (SELECT) من الجدول المذكور فى عبارة «من» (FROM). ويبين الشكل التالى تعليمة الاختيار فى أبسط صورها.

```
SELECT TableName.ColumnName, TableName.ColumnName, ...
FROM TableName;
```

مثـال ١: ما أرقام وأسماء المواد الدراسية التى تقدمها الجامعة الأهلية؟

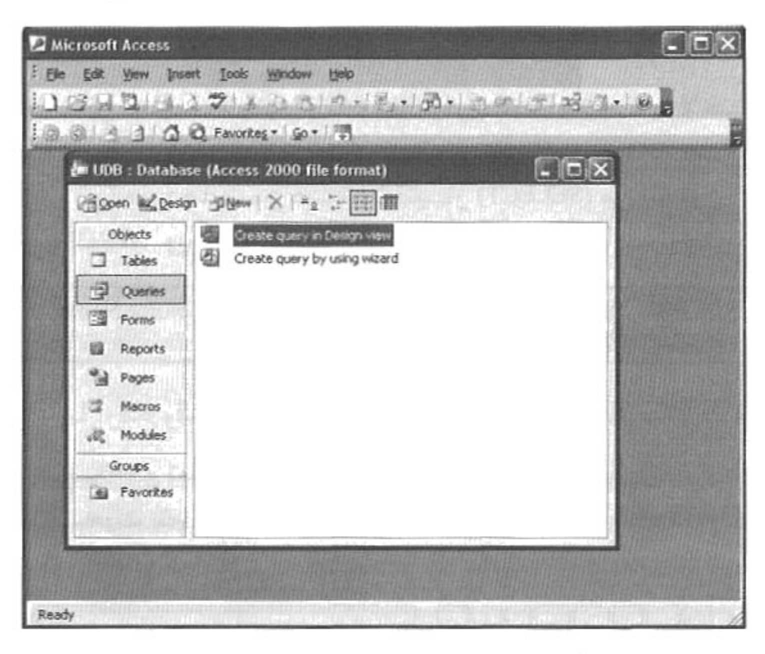
الحـل: نســتخدم تعليمـــة الاختيـــار (SELECT) على جـــدول المواد الدراســـية (COURSE_T) بحيث يتم اختيار حقل رمز المادة الدراسية (Course_ID) وحقل عنوان المادة الدراسية (Title) كما يلى:

```
SELECT COURSE_T.Course_Id, COURSE_T.Title
FROM COURSE_T;
```

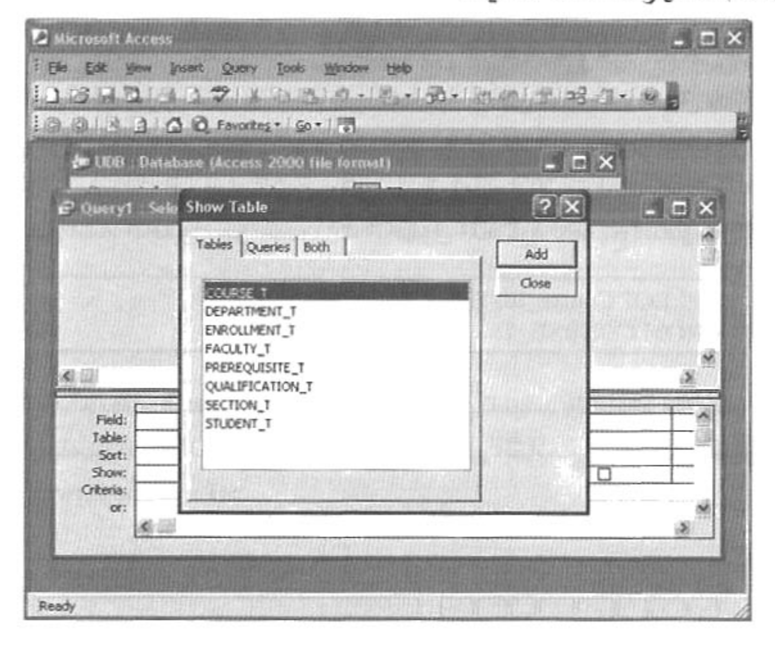
ويمكن الاســتغناء عن اســم الجدول فى تعليمه الاختيار (SELECT) (أو غيرها من تعليمـــات (SQL)) من أســـماء الحقول، وذلك فى حالة عدم وجود التباس فى أســـماء الحقول التابعة للجداول المســتخدمة فى التعليمة. ففى المثال الســابق جميع الحقول تتبـــع لجـــدول واحد، هو جدول المواد الدراســـية (COURSE_T)، لذلـــك فإنه لا يوجد التباس فى مســـميات الحقول التى سيتم تطبيق تعليمة الاختيار عليها، لذا فإنه يمكن الاستغناء عن اسم الجدول من مسميات الحقول المختارة لتصبح التعليمة كالتالى:

```
SELECT Course_Id, Title
FROM COURSE_T;
```

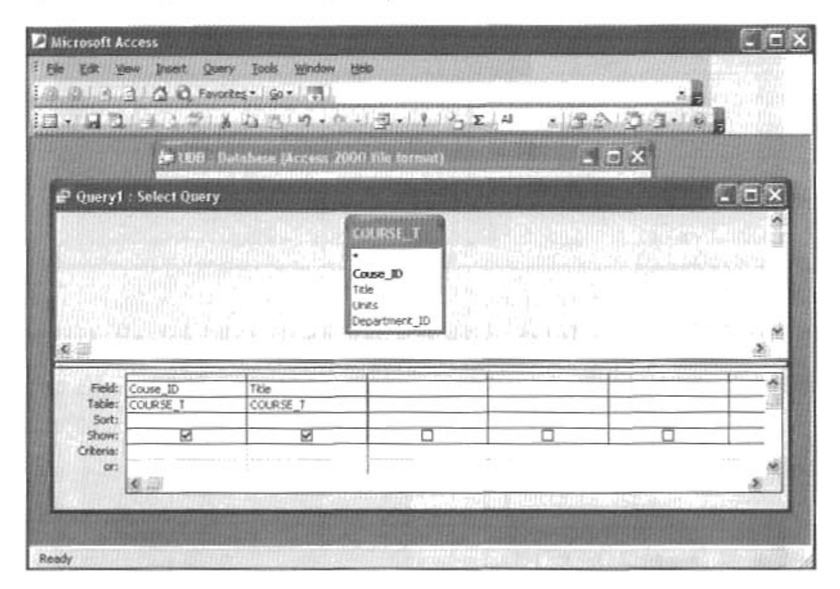
وباســتخدام نظام قاعدة بيانات أكســس يمكن تنفيــذ التعليمة من خلال الدخول لكينونة الاستفسارات (Query ICON) الموجودة ضمن القائمة الرئيسية للكينونات، كما يلى (Pargue and Irwin, 2001):



بعد ذلك يتم اختيار أيقونة إنشـــاء استفســـار فى وضع التصميم (Create query in design view) لتظهر الشاشة التالية:

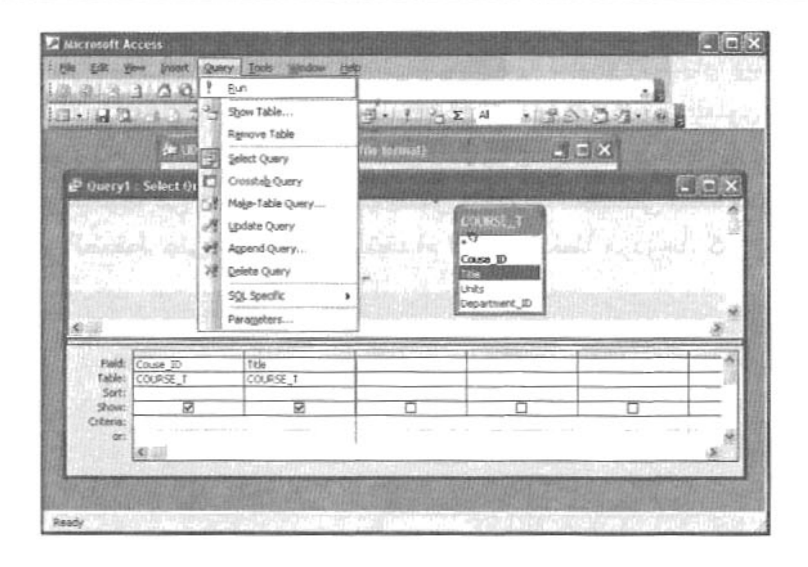


يتم اختيار جدول المواد الدراسية، الذى ستنفذ عليه تعليمة الاختيار، ويتم الضغط على أيقونة الإضافة (Add). عندئذ سـيتم إدراج الجدول فى الجزء العلوى من شاشة الاستفســار. بعد ذلـك يتم اختيار الحقلين المطلوبين، وذلك من خلال سـحبهما من الجـدول وإلقائهما فى الحقل المناســب ضمن المخطط السـفلى المخصص لإنشــاء التعليمة، أو الضغط على كل منهما باستخدام الفأرة ضغطاً مزدوجاً، كما يلى:

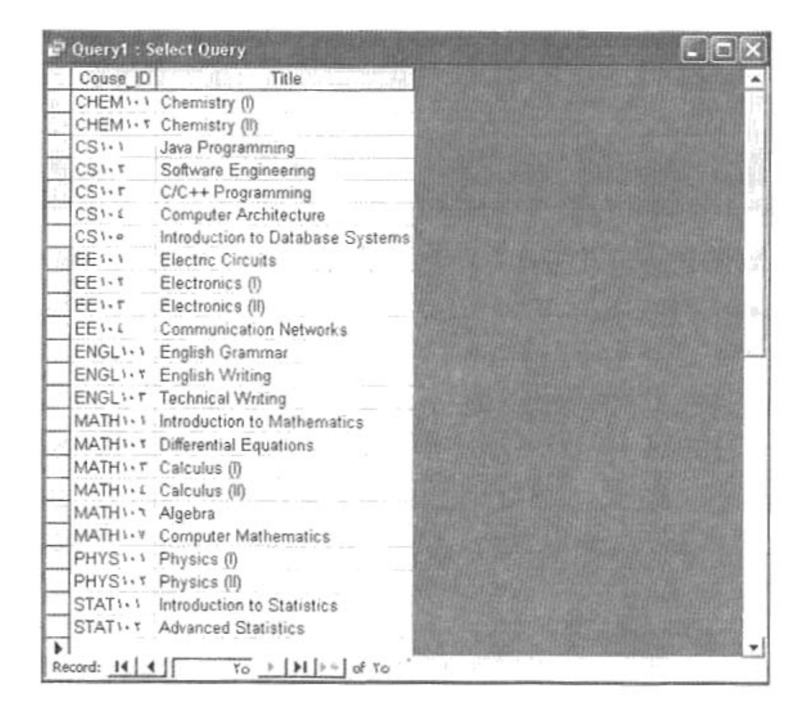


تجدر ملاحظة أن الحقل المسمى رمز المادة الدراسية (Course_ID) قد ظهر بالخط العريض للدلالة على أنه المفتاح الرئيسى للجدول. كما ظهر حقل جديد ضمن الجدول برمــز علامة النجمة (*). ويعنى هذا الحقل أنه ممثل لجميع حقول الجدول، بمعنى أنــه يمكن اختياره لإظهار كل حقول الجدول ضمن نتيجة عملية الاختيار دون الحاجة لسحب كل حقل على حدة وإلقائه ضمن مخطط إنشاء الاستفسار.

بعــد ذلك يتم تنفيذ عملية الاختيار من خــلال اختيار أمر التنفيذ (RUN) المدرجة ضمن قائمة الاستفســارات (Query) أو اختيار رمز تعليمة التنفيذ وهو علامة التعجب (!) من قائمة الاختصارات، كما يلى:



وبعد تنفيذ العملية تظهر نتيجة عملية الاختيار كما يلى:

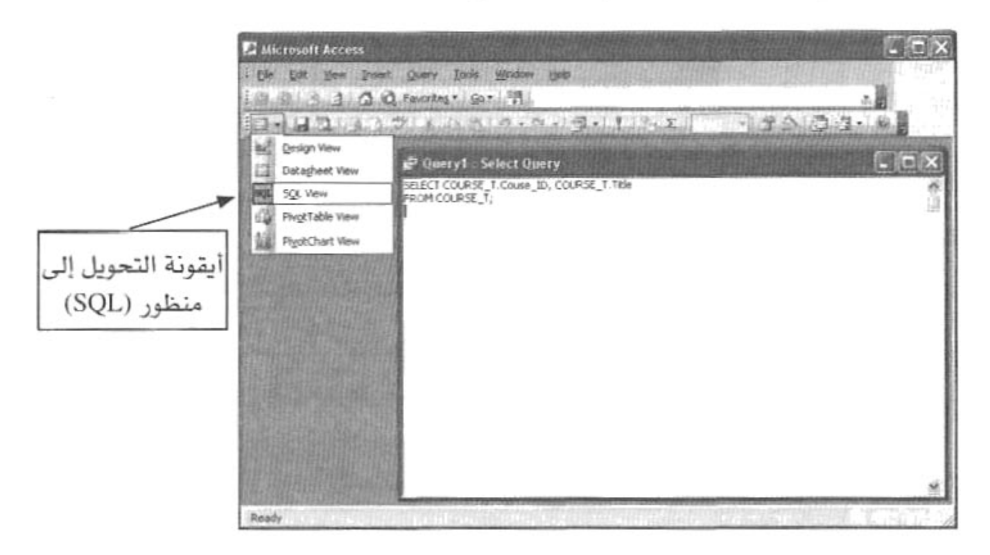


Query1 : Select Query

Couse_ID	Title
CHEM١٠١	Chemistry (I)
CHEM١٠٢	Chemistry (II)
CS١٠١	Java Programming
CS١٠٢	Software Engineering
CS١٠٣	C/C++ Programming
CS١٠٤	Computer Architecture
CS١٠٥	Introduction to Database Systems
EE١٠١	Electric Circuits
EE١٠٢	Electronics (I)
EE١٠٣	Electronics (II)
EE١٠٤	Communication Networks
ENGL١٠١	English Grammar
ENGL١٠٢	English Writing
ENGL١٠٣	Technical Writing
MATH١٠١	Introduction to Mathematics
MATH١٠٢	Differential Equations
MATH١٠٣	Calculus (I)
MATH١٠٤	Calculus (II)
MATH١٠٦	Algebra
MATH١٠٧	Computer Mathematics
PHYS١٠١	Physics (I)
PHYS١٠٢	Physics (II)
STAT١٠١	Introduction to Statistics
STAT١٠٢	Advanced Statistics

Record: ٢٥ of ٢٥

وتمكن قاعدة بيانات أكسس أيضاً من إطلاع المستخدم على تعليمة لغة الاستفسار البنائية المكافئة للتصميم الذى تم إعداده من خلال الانتقال إلى شاشة عرض (SQL). كما يمكن كتابة تعليمات (SQL) مباشرة من خلال شاشة عرض (SQL). والشكل التالى يوضح تعليمة (SQL) لحل المثال حسب ظهورها فى شاشة عرض (SQL) التابعة لقاعدة بيانات أكسس بعد عملية تصميم الاستفسار.

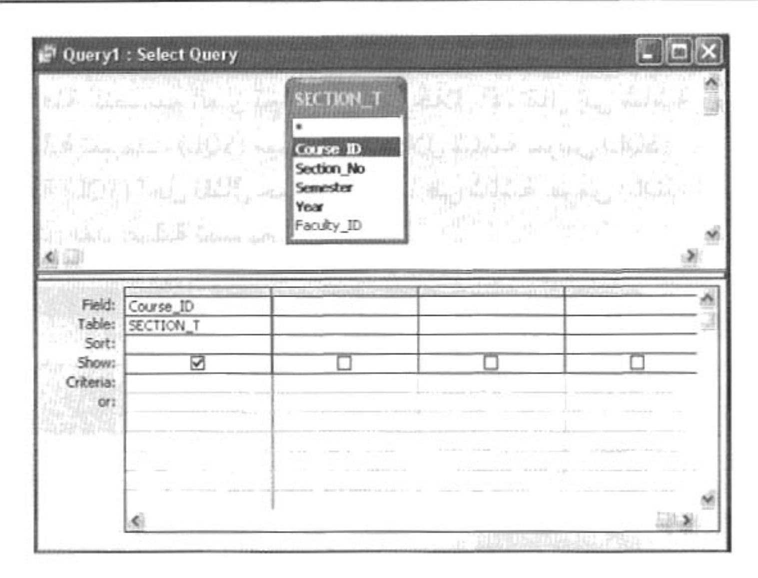


مثال ٢: ما أرقام المواد الدراسية المنفذة؟

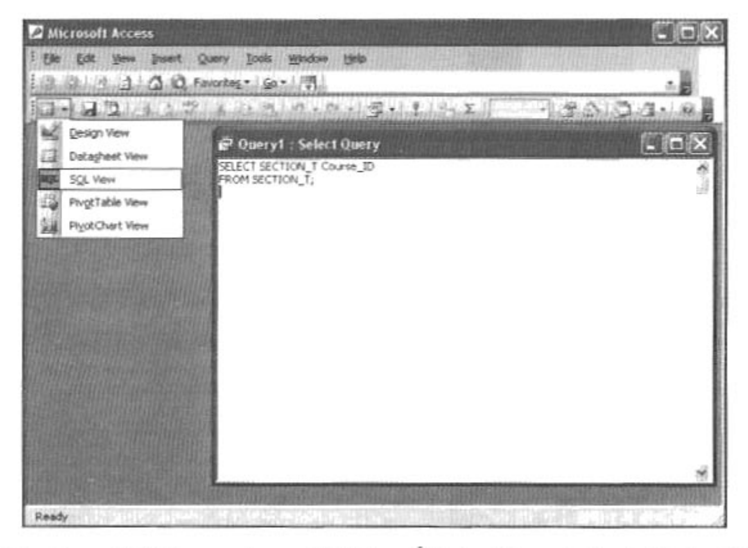
الحل: نستخدم تعليمة الاختيار (SELECT) مرةً أخرى ولكن على جدول المجموعات الدراسية (SECTION_T) عوضاً عن جدول المواد الدراسية (COURSE_T)، وبحيث يتم اختيار حقل رمز المادة الدراسية (Course_ID) كما يلى:

```
SELECT Course_ID
FROM SECTION_T;
```

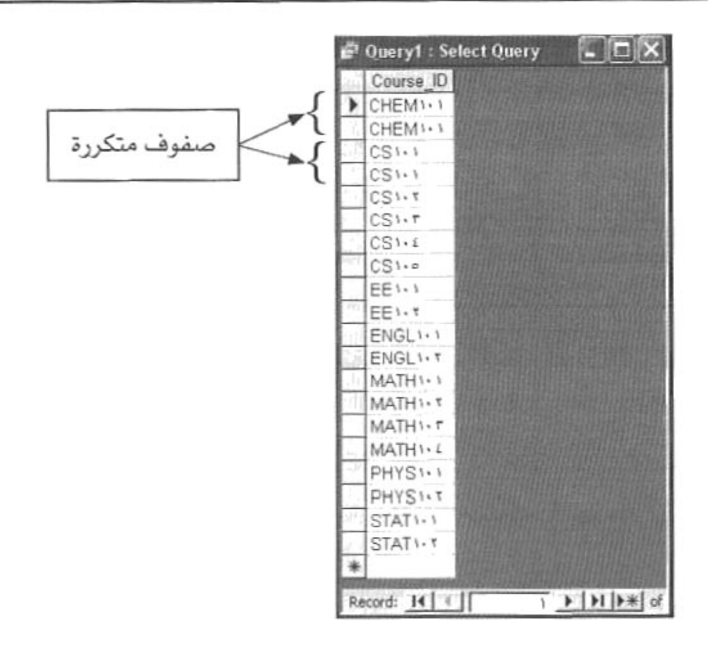
والشـــكل التالى يوضح شاشة عرض التصميم للتعليمة الســـابقة باستخدام نظام قاعدة بيانات أكسس.



وللاطـــلاع على شـــكل التعليمة من منظور (SQL) يتم الانتقال إلى شاشـــة عرض (SQL)، كما فى المثال السابق، ليظهر الشكل التالى للتعليمة.



وعند تنفيذ التعليمة باســتخدام الأمر (RUN)، سواء من شاشة عرض التصميم أو شاشة عرض (SQL) يكون الناتج كما هو موضح فى الجدول التالى.



يلاحظ فى نتيجة التعليمة تكرار مادة الكيمياء (١٠١) (CHEM101) ومادة الحاسب الآلى (١٠١) (CS101)، وذلك لكون كلتا المادتين ينفذان من خلال مجموعتين دراسيتين عوضاً عن مجموعة دراسـية واحدة. ويعنى هذا أن تعليمة الاختيار (SELECT) تقوم باختيار قيمة الحقل الذى تم تحديد اسـمه ضمن التعليمـة دون اعتبار لافتراض ما إذا كانت قيمته قد تم اختيارها سـابقاً ضمن نتائج التعليمة أم لا. إلا أنه فى الكثير مـن الأحيـان نحتاج إلـى إدراج قيم الصفوف فـى النتائج دون تكـرار. وتظهر هذه الحاجة بشكل خاص عندما لا يكون ضمن الحقول المحددة فى تعليمة الاختيار المفتاح الرئيسـى للجدول نظراً لإمكانية تكرار قيم بقية حقول الجدول، فى مثل هذه الحالة، كما يوضح المثال السابق. ولهذا السبب توفر مواصفات لغة الاستفسار البنائية (SQL) الكلمة المحجوزة (DISTINCT) التى تقوم بحذف الصفوف المتكررة من النتيجة النهائية للتعليمـة، وإظهار قيمة كل صف مـرة واحدة فقط بغض النظر عن المرات التى يتكرر فيها.

٧-٢-١-٢ حذف الصفوف المتكررة من نتيجة تعليمة الاختيار باستخدام كلمة (DISTINCT):

توفـر (SQL) الكلمة المحجـوزة (DISTINCT) ضمن تعليمة الاختيـار (SELECT). وتمكـن هذه الكلمة عند اسـتخدامها من حـذف تكرارات كل صـف وإظهاره ضمن

النتيجــة النهائية لتعليمة الاختيار مرة واحدة فقط بغض النظر عن عدد مرات تكرار الصف. وعند استخدام تعليمة الاختيار (SELECT) دون استخدام الكلمة (DISTINCT) فإن القيمة الافتراضية للتعليمة هى إظهار صفوف النتيجة كافة بما فيها المتكرر منها. والشكل التالى يوضح تعليمة الاختيار عند تضمينها على كلمة (DISTINCT).

```
SELECT DISTINCT ColumnName, ColumnName, . . .
FROM TableName;
```

مثـال ٣: ما أرقام المواد الدراســية المنفذة؟ أظهر أرقام المواد الدراسية دون تكرار (أى بغض النظر عن عدد المجموعات (أو الشعب) المنفذة من خلالها).

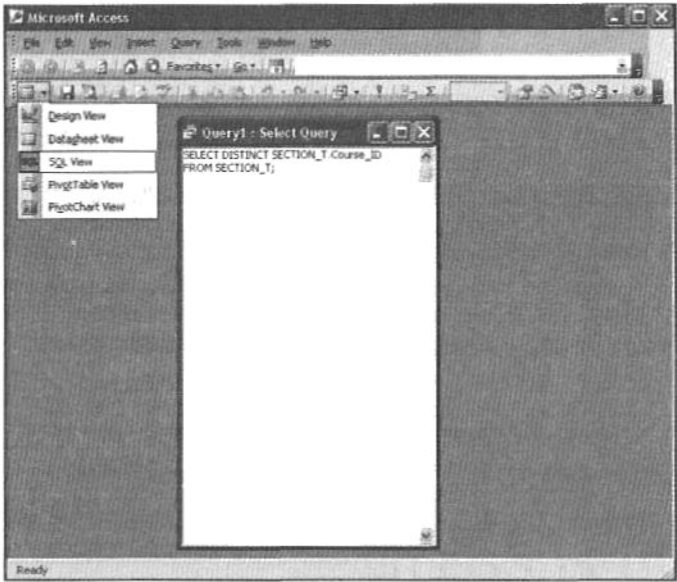
الحل: نســتخدم تعليمــة الاختيار (SELECT) مرةً أخرى علــى جدول المجموعات الدراســية (SECTION_T)، بحيث يتم اختيار حقل رمز المادة الدراسية (Course_ID) مع حذف الصفوف المتكررة من نتيجة الاختيار كما يلى:

```
SELECT DISTINCT Course_ID
FROM SECTION_T;
```

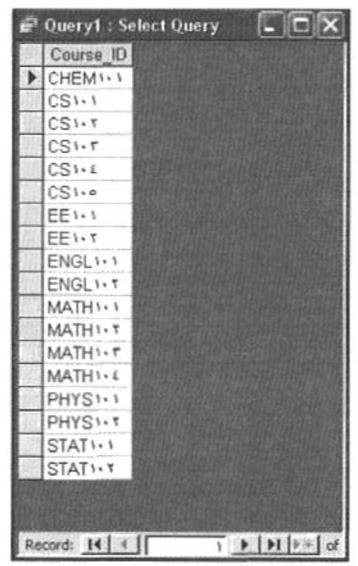
وتكون نتيجة التعليمة السابقة كما يلى:

```
COURSE_
-------
CHEM101
CS101
CS102
CS103
CS104
CS105
EE101
EE102
ENGL101
ENGL102
MATH101
MATH102
MATH103
MATH104
PHYS101
PHYS102
STAT101
STAT102
```

ويمكن تنفيذ التعليمة، فى بيئة أكسس، مباشرة من خلال شاشة عرض (SQL) كما يلى:



وتكون نتيجة التعليمة الجدول التالى الذى يلاحظ فيه حذف الصفوف المتكررة من نتيجة المثال السابق.



Course_ID
CHEM١٠١
CS١٠١
CS١٠٢
CS١٠٣
CS١٠٤
CS١٠٥
EE١٠١
EE١٠٢
ENGL١٠١
ENGL١٠٢
MATH١٠١
MATH١٠٢
MATH١٠٣
MATH١٠٤
PHYS١٠١
PHYS١٠٢
STAT١٠١
STAT١٠٢

كما يمكن حل المثال من خلال استخدام شاشة تصميم الاستفسار، بحيث يتم تغيير خواص الاستفسار (Properties) بعد تصميم الاستفسار ليظهر القيم بشكل غير متكرر حسب التالى:

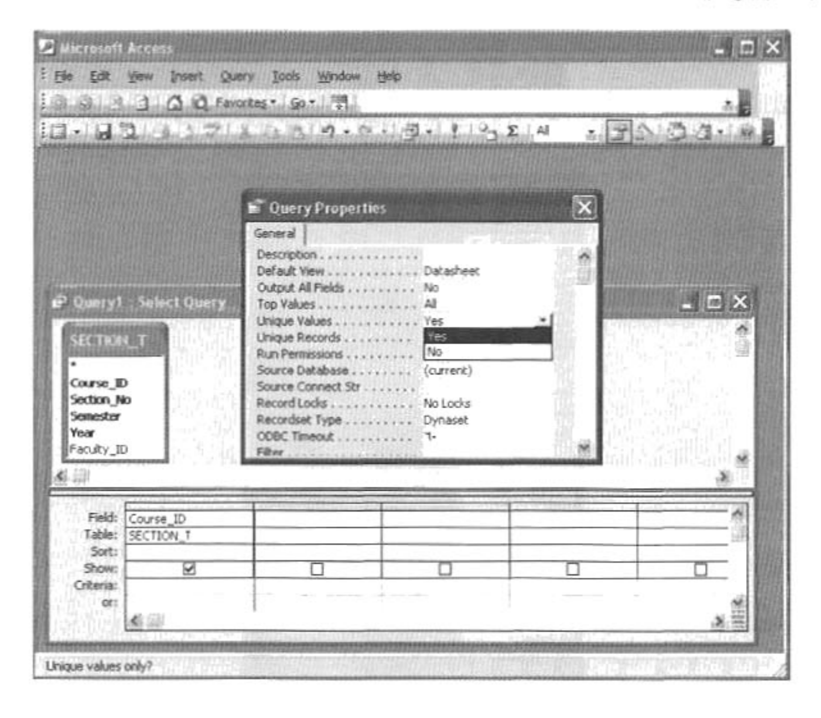
١- تصميم الاستفسار بحيث يظهر رمز البرنامج (Course_ID) من جدول (SECTION_T).

٢- الدخول إلى خصائص الاستفسار (Properties)، إما من خلال قائمة عرض أو بالضغط على زر الفأرة الأيمن (وذلك، عندما يكون المؤشر فى الجزء العلوى من شاشة التصميم) ومن ثم اختيار (Properties).

٣- تغيير قيمة حقل القيم المتكررة (Unique Records) من (No) إلى (Yes).

٤- تنفيذ الاستفسار.

ويوضح الشكل التالى شاشة عرض التصميم وقائمة الخواص موضحاً عليها الحقل الواجب تغيير قيمته.



ويمكـــن التأكد من أن تصميم التعليمة فى شاشـــة عرض التصميم مطابق لتعليمة (SQL) المتضمنة على كلمة (DISTINCT) من خلال الانتقال إلى شاشـــة عرض (SQL). كما أنه عند تنفيذ التعليمة من شاشـــة عرض التصميم تكون القيم المدرجة فى جدول النتيجة مطابقة لنتائج تنفيذ تعليمة (SQL) من شاشة عرض (SQL).

٧-٢-١-٣ الأسماء المستعارة للأعمدة (ALIAS):

قد تحتوى أســـماء الأعمدة فى الجداول على اختصارات بحيث يصعب على جميع المســـتخدمين لقاعدة البيانات فهمها، أو قد يكون من المتعارف عليه اســـتخدام اســـم معين من قبل مجموعة من المستخدمين لحقل معين يختلف عن الاسم الذى تم تعريفه به فى قاعدة البيانات. لذلك توفر لغة (SQL) القياســـية طريقة تمكن المســـتخدم من إعادة تسمية أعمدة الجداول عند عرض نتائج تعليمة الاختيار (SELECT) بحيث تظهر بمسميات مختلفة عن المسميات الأصلية التى سميت بها فى قاعدة البيانات. كما أن إعادة التســـمية هذه لا تؤثر فى المســـميات الأصلية للأعمدة، وإنما تتلاشى فاعليتها بمجرد إظهار نتائج التعليمة. ولإعادة تســـمية عمود ما بكلمة واحدة تتم كتابة الاسم الجديد له مباشـــرة بعد اسمه الأصلي. أما إذا كان الاسم الجديد مكوناً من أكثر من كلمة فإن الاســـم الجديد يكتب داخل علامتى تنصيص مزدوجة. والمثال التالى يوضح طريقة استخدام إعادة التسمية.

مثـال ٤: ما أرقام وأسماء المواد الدراسية التى تقدمها الجامعة الأهلية؟ اجعل اسم عمود أرقام المواد الدراســـية بمســـمى (#CRS) عوضاً عن مسماه الأصلى (Course_ID)، واسم عمود المواد الدراسية (Course Title) عوضاً عن مسماه الأصلى (Title).

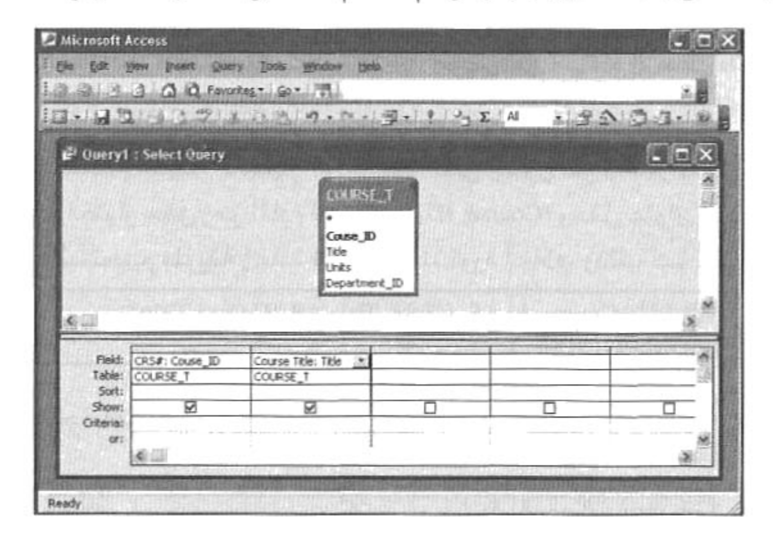
الحل: نستخدم تعليمة الاختيار (SELECT) على جدول المواد الدراسية (COURSE_T)، بحيث يتم اختيار حقل رمز المادة الدراسية (Course_ID) وحقل عنوان المادة الدراسية (Title). كما نستخدم طريقة إعادة التسمية المذكورة أعلاه، وذلك كما يلى:

```
SELECT Course_Id AS  CRS#, Title AS "Course Title"
FROM COURSE_T;
```

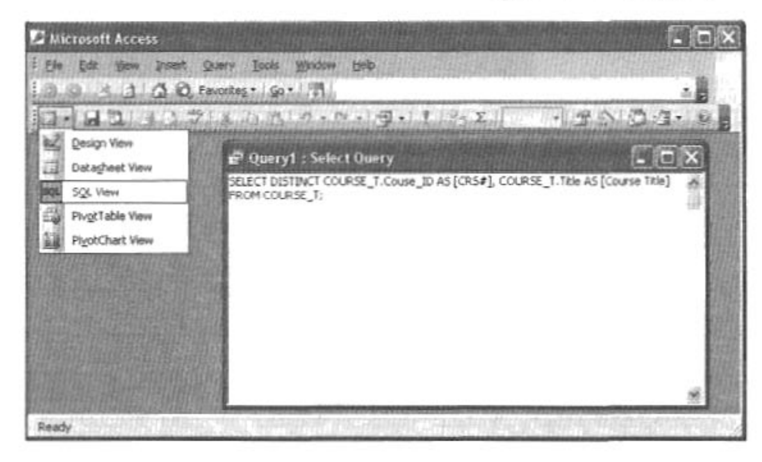
وتكون نتيجة التعليمة السابقة كما يلى:

```
CRS#    Course Title
------- ------------------------------------
CHEM101 CHEMISTRY (I)
CHEM102 CHEMISTRY (II)
CS101   JAVA PROGRAMMING
CS102   SOFTWARE ENGINEERING
CS103   C/C++ PROGRAMMING
CS104   COMPUTER ARCHITECTURE
CS105   INTRODUCTION TO DATABASE SYSTEMS
EE101   ELECTRIC CIRCUITS
EE102   ELECTRONICS (I)
EE103   ELECTRONICS (II)
EE104   COMMUNICATION NETWORKS
ENGL101 ENGLISH GRAMMAR
ENGL102 ENGLISH WRITING
ENGL103 TECHNICAL WRITING
MATH101 INTRODUCTION To MATHEMATICS
MATH102 DIFFERENTIAL EQUATIONS
MATH103 CALCULUS (I)
MATH104 CALCULUS (II)
MATH106 ALGEBRA
MATH107 COMPUTER MATHEMATICS
PHYS101 PHYSICS (I)
PHYS102 PHYSICS (II)
STAT101 INTRODUCTION TO STATISTICS
STAT102 ADVANCED STATISTICS
```

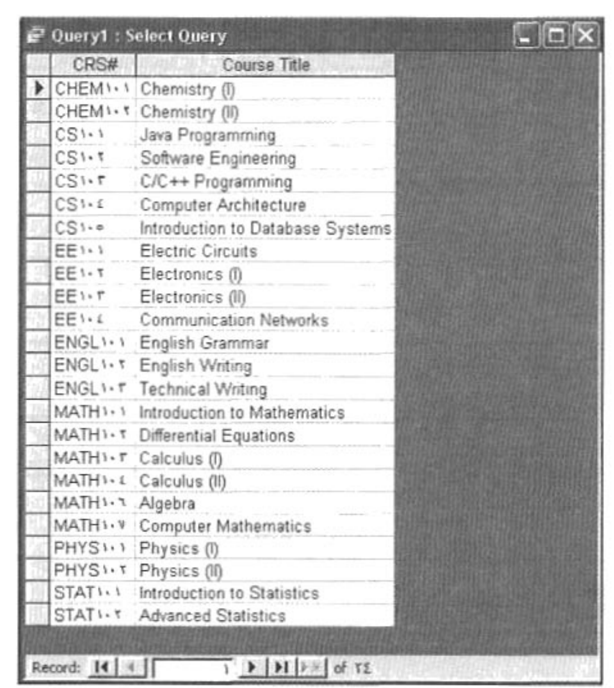
ويمكن تنفيذ التعليمة، فى نظام أكســس، من خلال كتابة المســمى الجديد للحقل متبوعاً بالنقطتين المتعامدين (:) ومن ثم الاسم الأصلى للحقل، كما يلى:



وتظهر تعليمة لغة الاستفســـار البنائية المصممة من خلال شاشــة عرض التصميم فى شاشة عرض (SQL) كما يلى:



أمـــا نتيجة تنفيذ التعليمـــة التى يلاحظ فيها تغير مســـميات الحقول، فتكون كما يلى:


Query1 : Select Query

CRS#	Course Title
CHEM١٠١	Chemistry (I)
CHEM١٠٢	Chemistry (II)
CS١٠١	Java Programming
CS١٠٢	Software Engineering
CS١٠٣	C/C++ Programming
CS١٠٤	Computer Architecture
CS١٠٥	Introduction to Database Systems
EE١٠١	Electric Circuits
EE١٠٢	Electronics (I)
EE١٠٣	Electronics (II)
EE١٠٤	Communication Networks
ENGL١٠١	English Grammar
ENGL١٠٢	English Writing
ENGL١٠٣	Technical Writing
MATH١٠١	Introduction to Mathematics
MATH١٠٢	Differential Equations
MATH١٠٣	Calculus (I)
MATH١٠٤	Calculus (II)
MATH١٠٦	Algebra
MATH١٠٧	Computer Mathematics
PHYS١٠١	Physics (I)
PHYS١٠٢	Physics (II)
STAT١٠١	Introduction to Statistics
STAT١٠٢	Advanced Statistics

Record: ١ of ٢٤

كما يمكن استخدام الأسماء المستعارة للجداول أيضاً فى تعليمة الاختيار، وخاصة عندما يستخدم الجدول نفسه أكثر من مرة فى نفس التعليمة لإجراء عملية ربط (Join Operation) كما سنرى لاحقاً عند شرح عملية الربط. والمثال التالى يوضح طريقة استخدام الأسماء المستعارة للجداول.

```
SELECT C.Course_Id, C.Title
FROM COURSE_T  C;
```

ويلاحظ فى المثال السابق أنه قد تم استخدام اسم مستعار لجدول المواد الدراسية (COURSE_T) فى عبارة مصدر الاختيار (FROM) ليصبح اسمه "C". وبعد ذلك، تم استخدام المسمى المستعار للجدول ضمن عبارة الاختيار (SELECT) فى تعليمة الاختيار. وعلى الرغم من أن عبارة مصدر الاختيار (FROM) تأتى بعد عبارة الاختيار، إلا أن هذا الترتيب لا يعنى أن تعليمة الاختيار تنفذ فى نظام قاعدة البيانات حسب تسلسل العبارات فى التعليمة كما سنوضح لاحقاً.

٧-٢-١-٤ اختيار كافة أعمدة جدول:

لاختيار جميع أعمدة جدول ما يمكن استخدام علامة النجمة (*) مع تعليمة الاختيار. وتقدم هذه الطريقة اختصاراً يعفينا عن سرد أسماء كل حقول الجدول. والشكل التالى يمثل تعليمة الاختيار عند استخدام هذه الطريقة:

```
SELECT *
FROM TableName;
```

أو

```
SELECT DISTINCT *
FROM TableName;
```

مثال ٥: ما تفاصيل جميع البرامج التدريسية التى توفرها الجامعة الأهلية؟

الحل:

```
SELECT *
FROM COURSE_T;
```

وتكون نتيجة التعليمة السابقة كما يلى:

```
COURSE_ TITLE                                          UNITS DEPART
------- ---------------------------------- ---------- ------
CHEM101 CHEMISTRY (I)                                      3 CHEM
CHEM102 CHEMISTRY (II)                                     3 CHEM
CS101   JAVA PROGRAMMING                                   3 CS
CS102   SOFTWARE ENGINEERING                               3 CS
CS103   C/C++ PROGRAMMING                                  3 CS
CS104   COMPUTER ARCHITECTURE                              3 CS
CS105   INTRODUCTION TO DATABASE SYSTEMS                   3 CS
EE101   ELECTRIC CIRCUITS                                  3 EE
EE102   ELECTRONICS (I)                                    3 EE
EE103   ELECTRONICS (II)                                   3 EE
EE104   COMMUNICATION NETWORKS                             4 EE
ENGL101 ENGLISH GRAMMAR                                    2 ENGL
ENGL102 ENGLISH WRITING                                    3 ENGL
ENGL103 TECHNICAL WRITING                                  3 ENGL
MATH101 INTRODUCTION To MATHEMATICS                        3 MATH
MATH102 DIFFERENTIAL EQUATIONS                             3 MATH
MATH103 CALCULUS (I)                                       3 MATH
MATH104 CALCULUS (II)                                      3 MATH
MATH106 ALGEBRA                                            4 MATH
MATH107 COMPUTER MATHEMATICS                               3 MATH
PHYS101 PHYSICS (I)                                        3 PHYS
PHYS102 PHYSICS (II)                                       3 PHYS
STAT101 INTRODUCTION TO STATISTICS                         3 STAT
STAT102 ADVANCED STATISTICS                                3 STAT
```

أمـــا فى حالة الرغبة فى ترتيب حقول الجدول ترتيباً معيناً، فإنه لا بد من ســـرد أســـماء حقول الجدول كافة ضمن تعليمة الاختيار وفقاً للترتيب الذى نرغب فيه؛ لأن اســـتخدام علامة النجمة ضمن تعليمة الاختيار تسترجع بيانات الحقول وفق الترتيب الذى عرفت فيه ضمن تعليمة إنشـــاء الجدول. فعلى ســـبيل المثال، لو أردنا استرجاع بيانات جدول المواد الدراســـية وفق الترتيب (Course_ID, Units, Department_ID, Title)، فإنه يمكن استخدام تعليمة الاختيار كما يلى:

```
SELECT COURSE_ID, UNITS,  DEPARTMENT_ID, TITLE
FROM COURSE_T;
```

وتكون نتيجة التعليمة السابقة كما يلى:

```
COURSE_       UNITS DEPART TITLE
------- ---------- ------ ------------------------------------
CHEM101          3 CHEM   CHEMISTRY (I)
CHEM102          3 CHEM   CHEMISTRY (II)
CS101            3 CS     JAVA PROGRAMMING
CS102            3 CS     SOFTWARE ENGINEERING
CS103            3 CS     C/C++ PROGRAMMING
CS104            3 CS     COMPUTER ARCHITECTURE
CS105            3 CS     INTRODUCTION TO DATABASE SYSTEMS
EE101            3 EE     ELECTRIC CIRCUITS
EE102            3 EE     ELECTRONICS (I)
EE103            3 EE     ELECTRONICS (II)
EE104            4 EE     COMMUNICATION NETWORKS
ENGL101          2 ENGL   ENGLISH GRAMMAR
ENGL102          3 ENGL   ENGLISH WRITING
ENGL103          3 ENGL   TECHNICAL WRITING
MATH101          3 MATH   INTRODUCTION To MATHEMATICS
MATH102          3 MATH   DIFFERENTIAL EQUATIONS
MATH103          3 MATH   CALCULUS (I)
MATH104          3 MATH   CALCULUS (II)
MATH106          4 MATH   ALGEBRA
MATH107          3 MATH   COMPUTER MATHEMATICS
PHYS101          3 PHYS   PHYSICS (I)
PHYS102          3 PHYS   PHYSICS (II)
STAT101          3 STAT   INTRODUCTION TO STATISTICS
STAT102          3 STAT   ADVANCED STATISTICS
```

٧-٢-١-٥ الاسترجاع المشروط (Conditional Retrieval):

تسترجع البيانات عادة باستخدام تعليمة الاختيار وفق شروط محددة. وتوفر لغة الاستفســار البنائية العديد من عوامل المقارنة المنطقية التى يمكن استخدامها ضمن عبارة شــرط الاسترجاع (WHERE). ومن عوامل المقارنة المنطقية المستخدمة فى لغة الاستفسار البنائية ما يلى:

عوامل المقارنة المستخدمة فى شرط الاسترجاع	
=	يساوى
<>	لا يساوى
>	أكبر من
>=	أكبر من أو يساوى
<	أقل من
<=	أقل من أو يساوى

وتأتى عبارة شرط الاسترجاع (WHERE) بعد عبارة مصدر الاسترجاع (FROM) فى الترتيب ضمن تعليمة الاختيار. والشكل التالى يوضح عبارة شرط الاسترجاع ضمن تعليمة الاختيار.

```
SELECT [*] | ColumnName1, ColumnName2, .... ColumnNameN
FROM TableName
WHERE  [ NOT ] ColumnName  Comparison-Operator  Literal;
```

ويقصد بعامل المقارنة (Comparison_Operator) فى تعليمة الاختيار السابقة أحد عوامل المقارنة المدرجة فى الجدول أعلاه. أما كلمة النفى (NOT) فهى اختيارية، ويعنى بها نفى شرط الاختيار الذى يليها، على حين يقصد بثابت المقارنة (Literal) إما قيمة ثابتة وإما مسمى حقل آخر. وعادة ما تتطلب قواعد المقارنة المنطقية توافق أنواع البيانات للحقول التى تقوم بالمقارنة بينها: مثل مقارنة رقمى برقمى أو سلسلة حرفية بسلسلة حرفية. كما يلاحظ أن البيانات الرقمية من نوع البيانات SMALLINT متوافق مع نوع البيانات NUMBER، والسلاسل الحرفية من نوع بيانات CHAR متوافق مع نوع البيانات VARCHAR أيضاً. وبذلك يعتمد نوع بيانات ثابت المقارنة على نوع بيانات العمود الذى تتم معه المقارنة. وفى حال كان ثابت المقارنة ثابتاً حرفياً فإنه يوضع بين علامتى تنصيص مفردتان (' ') إذا كانت بيانات العمود الخاضع للمقارنة حرفياً. كما يمكن أيضاً استخدام اسم عمود آخر أو تعبير حسابى بدلاً من استخدام قيمة ثابتة لثابت المقارنة كما سيتضح فيما بعد. وتمكن لغة الاستفسار البنائية من الربط بين أكثر من شرط استرجاع باستخدام العوامل المنطقية «و» (AND) و «أو» (OR).

مثال ٦: ما أسماء أعضاء هيئة التدريس الذين يعملون فى قسم اللغة الإنجليزية (ENGL)؟

الحل:

تطبق تعليمة الاختيار على جدول أعضاء هيئة التدريس بحيث يتم اختيار حقل الاسم الأول واسم العائلة من حقول الجدول، وبحيث يطبق شرط الاختيار على حقل رمز القسم الذى يجب أن يكون مساوياً للقيمة الثابتة ('ENGL').

```
SELECT  FNAME, LNAME
FROM  FACULTY_T
WHERE  DEPARTMENT_ID = 'ENGL';
```

وتكون نتيجة التعليمة السابقة الجدول التالى الذى يحتوى على أسماء أعضاء هيئة التدريس الذين يعملون فى قسم اللغة الإنجليزية:

```
FNAME        LNAME
------------ ------------
Yahya        Khorshid
Salem        Alhamad
Salman       Albassam
```

مثال ٧: ما أسماء ورواتب أعضاء هيئة التدريس الذين يتقاضون مرتبات تساوى ٤٠,٠٠٠ ألفاً أو أكثر؟

الحل:

```
SELECT  FNAME, LNAME, SALARY
FROM  FACULTY_T
WHERE  SALARY >= 40000 ;
```

وتكون نتيجة التعليمة السابقة كما يلى:

```
FNAME        LNAME            SALARY
------------ ------------ ----------
Mohammed     Alhamad           44000
Ghanim       Alghanim          44500
Saleh        Alghamdi          44600
Salem        Alhamad           40000
Fahad        Alzaid            44300
Saud         Alkhalifa         44900
Sultan       Aljasir           43300
Ali          Albader           45300
Saad         Alzhrani          44200
```

مثال ٨: ما أسماء ورواتب أعضاء هيئة التدريس الذين لا يعملون فى قسم الحاسب الآلى 'CS'؟

الحل:

```
SELECT FNAME, LNAME, SALARY
FROM FACULTY_T
WHERE DEPARTMENT_ID < > 'CS' ;
```

أو

```
SELECT FNAME, LNAME, SALARY
FROM FACULTY_T
WHERE NOT DEPARTMENT_ID = 'CS' ;
```

وتكون نتيجة التعليمتين السابقتين كما يلى:

```
FNAME        LNAME            SALARY
------------ ------------ ----------
Khalid       Aloufi            35000
Fahad        Alhamid           25900
Ahmad        Alotaibi          33900
Saleh        Alghamdi          44600
Yahya        Khorshid          36700
Salem        Alhamad           40000
Salman       Albassam          33800
Turki        Alturki           27800
Fahad        Alzaid            44300
Saud         Alkhalifa         44900
Mahmood      Alsalem           31900
Mishal       Almazid           29800
Sultan       Aljasir           43300
Ali          Albader           45300
Saad         Alzhrani          44200
Ahmad        Alsabti           33900
```

مثال ٩: ما أسماء ورواتب أعضاء هيئة التدريس الذين يعملون فى قسم الحاسب الآلى، وتزيد رواتبهم عن ٤٠,٠٠٠ أو تقل عن ٣٠,٠٠٠؟

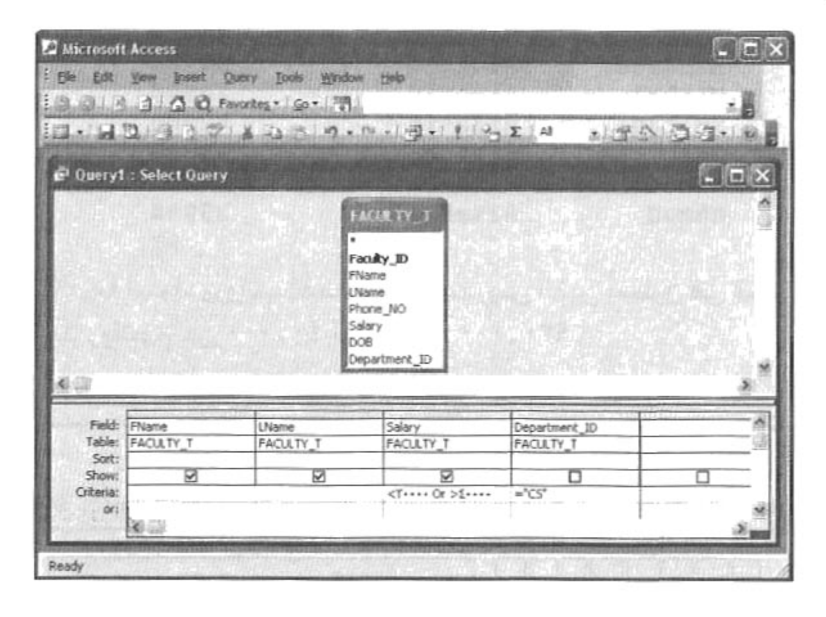
الحل:

```
SELECT FNAME, LNAME, SALARY
FROM FACULTY_T
WHERE DEPARTMENT_ID = 'CS' AND (SALARY > 40000 OR SALARY < 30000);
```

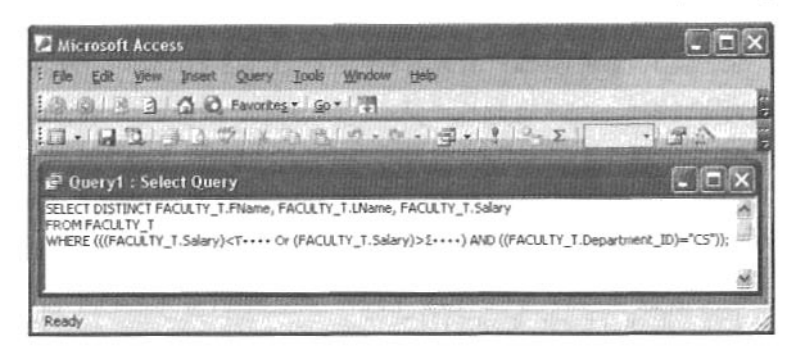
وتكون نتيجة التعليمة السابقة كما يلى:

```
FNAME        LNAME        SALARY
------------ ------------ ----------
Mohammed     Alhamad           44000
Ghanim       Alghanim          44500
Ibraheem     Alsaleh           25000
```

كما يمكن حل المثال السابق فى بيئة أكسس من خلال وضع شروط الاسترجاع فى الصف المعنون بالمعيار (Criteria) والصف الذى يليه والمعنون (OR)، حيث يمكن وضع الشروط فى هذين الصفين حسب الروابط المنطقية بين الشروط المختلفة، سواء كانت «و» أم «أو». فإذا أردنا الربط بين الشروط بالرابط المنطقى «و» فإننا نضع الشروط فى الصف نفسه المخصص للمعيار (Criteria). أما إذا أردنا أن نربط بين الشروط بالرابط المنطقى «أو» فإننا نضعها فى الصف المعنون (OR). ويمكن الربط بتوليفات مختلفة من العوامل المنطقية بين الشروط المختلفة وفق ذلك كما يوضح هذا المثال. كما يوضح هذا المثال أيضاً استخدام الحقل (Department_ID) من جدول أعضاء هيئة التدريس ووضع شرط عليه دون إظهاره ضمن نتيجة التعليمة، وذلك من خلال عدم اختياره بكلمة إظهار (Show). ويعنى هذا أن هذا الحقل لن يظهر ضمن أعمدة نتيجة العملية.



ويمكن الانتقال لشاشـــة عرض (SQL) لرؤية تعليمة لغة الاستفســـار البنائية التى تتوافق مع تصميم التعليمة السابق والتى تكون كما يلى:



٧-٢-١-٥-١ العوامل العلاقية (LIKE, BETWEEN and IN):

يستعرض هذا الجزء من الفصل ثلاثة عوامل علاقية أخرى يمكن استخدامها فى شرط الاسترجاع (WHERE) هى العامل العلاقى «مثل» (LIKE) الذى يستخدم لتحديد مواصفات معينة للسلســـلة الحرفية التى يجـــب أن تنطبق عليها قيمة حقل حرفى ما ضمن عبارة شـــرط الاسترجاع حتى يتم اختيار السجل الذى ينتمى إليه الحقل ضمن نتيجة تعليمة الاختيار، والعامل «بين» (BETWEEN) الذى يحدد مدى معين لقيمة حقل ما ضمن عبارة شـــرط الاسترجاع بحيث يجب أن تكون قيمة الحقل ضمن المدى الذى تم تحديده حتى يظهر الســـجل الذى ينتمى إليه الحقل ضمن نتيجة تعليمة الاختيار، والعامـــل «فى» (IN) الذى يحـــدد مجموعة من القيم يجب أن تكـــون قيمة الحقل من ضمنها حتى يظهر السجل الذى يتبعه الحقل ضمن نتيجة تعليمة الاختيار.

٧-٢-١-٥-١-١ العامل العلاقى «مثل» (LIKE Comparison Operator):

يستخدم العامل العلاقى «مثل» (LIKE) ضمن شرط الاسترجاع (WHERE) للمقارنة بين قيمة حقل ما بياناته من نوع السلاسل الحرفية بقيمة جزئية ثابتة لسلسلة حرفية ما، بحيث يجب أن تتضمن قيمة الحقل قيمة السلســـلة الحرفية الجزئية الثابتة حتى يتحقق شرط الاختيار ويظهر السجل الذى يتبعه الحقل ضمن نتيجة تعليمة الاختيار. ويمكن أن تتكون السلسلة الحرفية الجزئية من أية حروف وأرقام وحروف خاصة هى: (!#@$^&*-+=) بالإضافة للرموز الشاملة (Wildcards) المستخدمة مع LIKE وهى:

- الرمز الشامل (%) ويستخدم ليحل محل صفر أو أكثر من الحروف فى السلسلة الحرفيــة الجزئيــة. فمثلاً عند البحث عن الأســماء التى تبــدأ بالحرف (G) وتنتهــى بالحــرف (m) بغض النظر عــن الحروف بينهما يمكن كتابة السلســة الحرفيــة الجزئية كما يلى: (LIKE 'G%m'). أما إذا أردنا البحث عن الأســماء التــى تبدأ بالحرف (A) بغــض النظر عما يليها من حــروف، فإنه يمكن كتابة السلسلة الجزئية كما يلى: (LIKE 'A%') وإذا ما أردنا تلك التى تنتهى بالحرف (m) بغض النظر عما يســبقها من حروف، يمكن كتابة السلســلة الحرفية كما يلى: (LIKE '%m').

- الرمز الشــامل (_) ويســتخدم ليحل محل حرف واحــد فقط من الحروف فى السلســلة الحرفية الجزئيــة وفى مكان محدد فيها. فعلى ســبيل المثال يمكن البحث عن الأسماء ذات طول ستة أحرف وتنتهى بالحرف (m) كما يلى: (LIKE '_____m')، بحيث إن تكرار الرمز الشامل فى السلسة الحرفية الجزئية السابقة هو خمس مرات.

ويمكن اســتخدام الرمزين الشــاملين الســابقين فى توليفات مختلفة ضمن عامل المقارنة «مثل» (LIKE). فمثلاً عند البحث عن الأسماء التى تبدأ بالحرف (A) متبوعاً بحــرف واحد فقط فى الموقــع الثانى بغض النظر عن الحــرف الثاني، وبحيث يكون فيهــا الحرف الثالث هو (m)، وبغــض النظر عما يتبع الحرف الثالث من حروف، فإنه يمكن كتابة السلسة الحرفية الجزئية كما يلى: (LIKE 'A_m%'). وتجدر الملاحظة بأن حجم الحرف المســتخدم، سواء كان صغيراً أم كبيراً، ضمن أية سلسلة حرفية فى لغة الاستفسار البنائية له أهميته. فعلى سبيل المثال يعد الحرف (A) والحرف (a) مختلفين نهائياً، وليس بينهما أية علاقة عند اســتخدامنا لهما ضمن السلاسل الحرفية، سواء كانت ضمن عامل المقارنة «مثل» أو فى أى موقع تستخدم فيه السلاسل الحرفية.

ويمكن اســتخدام عامل النفــى (NOT) بالإضافة لعامل المقارنــة «مثل» بحيث يتم اختيار جميع الســجلات التى لا ينطبق عليها عامل المقارنة «مثل». وفيما يلى الشكل العام لعامل المقارنة «مثل».

```
SELECT ColumnName(s)
FROM TableName
WHERE ColumnName [NOT] LIKE Matching_sub_String ;
```

مثال:

ما أســماء أعضاء هيئة التدريس التى فيها الحرفان (ha) فى الموقع الثالث والرابع من أسماء عائلاتهم، على التوالى، بغض النظر عما يسبق ذلك من حروف أو يتبعه؟.

الحل:

```
SELECT FNAME, LNAME
FROM  FACULTY_T
WHERE  LNAME LIKE '__ha%';
```

وتكون النتيجة التعليمة كما يلى:

```
FNAME        LNAME
------------ ------------
Fahad        Alhamid
Mohammed     Alhamad
Salem        Alhamad
```

٧-٢-١-٥-١-٢ العامل العلاقى «بين» (BETWEEN Comparison Operator):

يســتخدم العامل العلاقى «بين» (BETWEEN) ضمن شــرط الاسترجاع (WHERE) لتحديد مدى معين لقيمة حقل ما ضمن عبارة شرط الاسترجاع، بحيث يجب أن تكون قيمة الحقل ضمن المدى الذى تم تحديده حتى يظهر الســجل الذى ينتمى إليه الحقل ضمن نتيجة تعليمة الاختيار. ويمكن اســتخدام العامــل العلاقى «بين» مع أى حقل، ســواء كانت بياناته من النــوع الرقمى، أو الحرفي، أو من نــوع التاريخ والوقت. كما يمكن استخدام عامل النفى (NOT) بالإضافة إلى عامل المقارنة «مثل» عند الرغبة فى اختيار جميع الســجلات التى لا ينطبق عليها عامل المقارنة «بين». وفيما يلى الشــكل العام لعامل المقارنة «بين».

```
SELECT ColumnName(s)
FROM TableName
WHERE ColumnName [NOT] BETWEEN Value1 AND Value2;
```

مثال:

ما أسماء أعضاء هيئة التدريس ومرتباتهم بين ٣٠,٠٠٠ ألفاً و ٤٠,٠٠٠ ألفاً؟

الحل:

```
SELECT FNAME, LNAME, SALARY
FROM FACULTY_T
WHERE SALARY BETWEEN 30000 AND 40000 ;
```

وتكون النتيجة التعليمة كما يلى:

FNAME	LNAME	SALARY
Khalid	Aloufi	35000
Saleh	Aleesa	30000
Ahmad	Alotaibi	33900
Yahya	Khorshid	36700
Salem	Alhamad	40000
Salman	Albassam	33800
Mahmood	Alsalem	31900
Ahmad	Alsabti	33900

مثال:

ما أسماء أعضاء هيئة التدريس ومرتباتهم أقل من ٣٠,٠٠٠ ألفاً أو أكثر ٤٠,٠٠٠ ألفاً؟

الحل:

```
SELECT FNAME, LNAME, SALARY
FROM FACULTY_T
WHERE SALARY NOT BETWEEN 30000 AND 40000 ;
```

وتكون النتيجة كما يلى:

FNAME	LNAME	SALARY
Fahad	Alhamid	25900
Mohammed	Alhamad	44000
Ghanim	Alghanim	44500
Ibraheem	Alsaleh	25000
Saleh	Alghamdi	44600
Turki	Alturki	27800
Fahad	Alzaid	44300
Saud	Alkhalifa	44900
Mishal	Almazid	29800
Sultan	Aljasir	43300
Ali	Albader	45300
Saad	Alzhrani	44200

مثال:

ما أســماء أعضاء هيئة التدريس الذين تبدأ أســماؤهم الأولى بالحروف التى تقع بين الحرف (A) والحرف (G)؟.

الحل:

```
SELECT FNAME, LNAME, SALARY
FROM  FACULTY_T
WHERE  FNAME BETWEEN 'A' AND 'G';
```

النتيجة:

```
FNAME        LNAME        SALARY
------------ ------------ ----------
Fahad        Alhamid           25900
Ahmad        Alotaibi          33900
Fahad        Alzaid            44300
Ali          Albader           45300
Ahmad        Alsabti           33900
```

ويلاحظ فى النتيجة السابقة أنها لا تتضمن عضو هيئة التدريس «غانم» (Ghanim)، وذلك لكون هذا الاســم يأتى فى الترتيب بعد الحرف (G) الذى تم تحديده فى شــرط الاسترجاع. وإذا ما أردنا إظهار الأسماء التى تبتدأ بالحرف "G" أيضا فيمكنا تحديد الحرف التالى للحرف (G) وهو الحرف (H).

مما ســبق يتضح أن العامل العلاقى «بين» يكافئ (≡) استخدام عامل المقارنة (=<) و عامــل المقارنة (=>) وذلــك عند ربطهما بالعامل المنطقى «و» (AND) ضمن شــرط الاسترجاع، كما يلى:

```
ColumnName BETWEEN Value1 AND Value2 ≡
ColumnName >= Value1 AND   ColumnName <= Value2
```

ويمكن حل المثال الســابق باســتخدام عامل المقارنة «أكبر من أو يســاوى» وعامل المقارنة «أصغر من أو يساوى»، كما يلى:

```
SELECT FNAME, LNAME, SALARY
FROM  FACULTY_T
WHERE  FNAME >=  'A' AND FNAME <= 'G';
```

٧-٢-١-٥-١-٣ العامل العلاقى «فى» (IN Comparison Operator):

يســتخدم العامل العلاقى «فى» (IN) ضمن شرط الإســترجاع (WHERE) لتحديد مجموعــة من القيم بحيث يجب أن تكون قيمة الحقل من ضمنها حتى يظهر الســجل الــذى يتبعه الحقل ضمن نتيجة تعليمة الاختيار. وقــد تكون مجموعة القيم ثابتة أو ناتجة من خــلال عمليات اختيار متداخلة أخرى كما ســنوضح فى الجزء (٨-٢-١). وكمــا هــو الحال فى العامل العلاقى «بــين»، يمكن أن تكون بيانــات الحقل من النوع الرقمى، أو الحرفى، أو نوع التاريخ والوقت. كما يمكن اســتخدام عامل النفى (NOT) بالإضافة لعامل المقارنة «فى» عند الرغبة فى اختيار جميع الســجلات التى لا ينطبق عليها عامل المقارنة «فى». وفيما يلى الشكل العام لعامل المقارنة «فى».

```
SELECT ColumnName(s)
FROM TableName
WHERE ColumnName [NOT] IN (Value1, Value2, ..., ValueN);
```

مثال:

ما أسماء أعضاء هيئة التدريس الذين رواتبهم ٤٠,٠٠٠ ألفاً أو ٤٤,٠٠٠ ألفاً؟

الحل:

```
SELECT FNAME, LNAME, SALARY
FROM  FACULTY_T
WHERE  SALARY IN (40000, 44000);
```

وتكون النتيجة كما يلى:

FNAME	LNAME	SALARY
Mohammed	Alhamad	44000
Salem	Alhamad	40000

مثال:

ما أســماء ورواتب أعضاء هيئة التدريس ذوى أســماء العائــلات غير (Alhamid)، و(Alotaibi) و(Alzaid) و(Albader) و(Alsabti).

الحل:

```
SELECT FNAME, LNAME, SALARY
FROM  FACULTY_T
WHERE  LNAME NOT IN ('Alhamid', 'Alotaibi', 'Alzaid', 'Albader', 'Alsabti');
```

وتكون النتيجة كما يلى:

```
FNAME        LNAME        SALARY
------------ ------------ ----------
Khalid       Aloufi            35000
Saleh        Aleesa            30000
Mohammed     Alhamad           44000
Ghanim       Alghanim          44500
Ibraheem     Alsaleh           25000
Saleh        Alghamdi          44600
Yahya        Khorshid          36700
Salem        Alhamad           40000
Salman       Albassam          33800
Turki        Alturki           27800
Saud         Alkhalifa         44900
Mahmood      Alsalem           31900
Mishal       Almazid           29800
Sultan       Aljasir           43300
Saad         Alzhrani          44200
```

وكما هو الحال بالنسبة للعامل العلاقى «بين»، يمكن استخدام عوامل مقارنة أخرى للحصول على ما يكافئ عامل المقارنة «فى». وعلى وجه التحديد يمكن استخدام عامل المقارنة (=) أكثر من مرة مربوطاً بالعامل المنطقى «أو» (OR) ضمن شرط الاسترجاع، كما يلى:

```
ColumnName IN (Value1, Value2, ..., ValueN) ≡
ColumnName = Value1 OR   ColumnName = Value2 . . . OR ColumnName = ValueN
```

ويمكن حل المثال السابق باستخدام عامل المقارنة «يساوى» وعامل المقارنة «أو»، كما يلى:

```
SELECT FNAME, LNAME, SALARY
FROM  FACULTY_T
WHERE NOT (LNAME =  'Alhamid'  OR LNAME = 'Alotaibi'
              OR LNAME =  'Alzaid' OR LNAME =  'Albader'
              OR LNAME = 'Alsabti');
```

٧-٢-١-٥-٢ القيم غير المعرفة (NULL VALUES):

تسمح لغة الاستفسار البنائية للحقول بأن تكون قيم بياناتها، وليس نوعية بياناتها، من النوع غير المعرف (NULL). ويمكن أن تفسر القيمة غير المعرفة (NULL) عند وجودها فى حقل ما وفق عدد من التفسيرات التى يأتى من ضمنها التفسيرات الثلاثة التالية التى تعد الأكثر شيوعاً.

- **قيمة الحقل غير متوافرة حالياً (Unavailable):** القيمة موجودة فعلياً على أرض الواقع ولكنها غير متوافرة حالياً، ومن ثم لا يمكن إدخالها إلى الحقل. ومن الأمثلة القابلة لهذا التفسير قيم حقل تاريخ الميلاد؛ إذ إن لكل شخص تاريخ ميلاد، وإن وجود قيمة غير معرفة فى حقل تاريخ الميلاد لشخص معين لا يعنى أنه لا يوجد لهذا الشخص تاريخ ميلاد، ولكن القيمة غير المعرفة في هذه الحالة تعنى عدم توافر تاريخ الميلاد لهذا الشخص فى الوقت الراهن ضمن بيانات قاعدة البيانات.
- **قيمة الحقل متحفظ عليها (Withheld):** القيمة موجودة فعلياً على أرض الواقع ولكنه تم التحفظ عليها. ومن أمثلة ذلك عدم تزويد أحد الموظفين لرقم هاتفه المنزلى بشكل مقصود على الرغم من وجود خط هاتف منزلى له، وذلك بغية التحفظ على رقم هاتفه وعدم الرغبة فى إفشائه.
- **قيمة الحقل غير منطبقة (Inapplicable):** لا توجد قيمة يمكن إدخالها إلى الحقل وتنطبق مع ما يتوافر على أرض الواقع. ومن أمثلة ذلك عدم توافر درجة علمية (بكالوريوس، ماجستير، دكتوراه) لأحد الموظفين بحيث يمكن إدخالها فى حقل الدرجة العلمية التابع لجدول الموظفين.

إن مبدأ القيم غير المعرفة مبدأ فعال جداً؛ إذ إنه يعفينا من تدوين الحالات السابقة بشكل واضح ضمن بيانات قاعدة البيانات. فعلى سبيل المثال، بدون هذا المبدأ فإنه يجب إدخال رمز خاص فى حقل تاريخ الميلاد لبيان أن تاريخ الميلاد غير متوافر حالياً مثل الرمز (12-12-1212)، أو رمز خاص لأرقام الهواتف المتحفظ عليها مثل (999-9999)، أو رمز خاص لحقل الدرجة العلمية غير المنطبق على الموظفين مثل (no_degree). إن إدخال مثل هذه الرموز ضمن بيانات قاعدة البيانات يعقد فهم محتويات قاعدة البيانات كما أنه يؤدى إلى صعوبة التعامل معها حيث يجب على جميع المستفيدين من قاعدة البيانات معرفة هذه الرموز ومعانيها. كما يعنى هذا تعقيد كتابة التطبيقات المبنية على قاعدة البيانات.

وعلى الرغم من فاعلية مبدأ القيم غير المعرفة إلا أنه يجب توخى الحذر الشديد عند التعامل معها من خلال لغة الاستفسار البنائية؛ لأنها تعد مصدراً للالتباس (Garcia-Molina et al, 2002). فعند استخدام أحد الحقول التى قد تحتوى على قيم غير معرفة ضمن شرط الاختيار (WHERE) فإنه يجب تذكر قاعدتين رئيسيتين:

- عندما نتعامل مع قيمة غير معرفة من خلال العوامل الحسابية، مثل الضرب والجمع، فإن نتيجة العملية الحسابية تكون قيمة غير معرفة أيضاً. فعلى سبيل المثال، لنفترض الحقل (X) التابع لأحد الجداول ونوع بياناته عددية (Number). ولنفترض أيضاً أن قيمة الحقل (X) فى أحد سجلات الجدول هو القيمة غير المعرفة (NULL). فإذا ما أجرينا العملية الحسابية (X+10) فإن نتيجة العملية ستكون غير معرفة (NULL). كذلك هو الحال لو أجرينا العملية (X*0) أو العملية (X-X) فإن النتيجة ستكون غير معرفة أيضاً على الرغم من أن ناتج عملية ضرب أى عدد بالعدد صفر هو صفر، وأن ناتج طرح أى عدد من نفسه هو صفر أيضاً! ويعنى هذا أن نتيجة أية عملية حسابية تتضمن حقلاً قيمته غير معرفة تكون نتيجته غير معرفة.

- عندما نتعامل مع قيمة غير معرفة من خلال عوامل المقارنة مثل (=، <>، ... إلخ)، فإن النتيجة تكون غير معلومة (UNKNOWN) (وليست غير معرفة). فعلى سبيل المثال، عندما نقارن حقل قيمته غير معرفة وليكن (X) أيضاً بالعدد ٣ مثل (X=3) أو (X>3)، فإن نتيجة عملية المقارنة ستكون غير معلومة (UNKNOWN).

وحيث إن عمليات المقارنة قد تقود إلى نتيجة غير معلومة (UNKNOWN)، فإن هذا يستدعى تعريف المنطق الثلاثى الذى يحتوى على القيمة غير المعلومة (UNKNOWN) بالإضافة للصح (True)، والخطأ (False)، وذلك حتى نتمكن من الربط بين هذه القيم من خلال العوامل المنطقية (Logical Operators) وهى «و» (AND)، «أو» (OR)، «النفى» (NOT).

٧-٢-١-٥-٢-١ المنطق الثلاثى القيم (Three-Valued Logic):

عند تعاملنا مع القيم غير المعرفة من خلال عوامل المقارنة قد ينتج عن ذلك قيم غير معلومة (UNKNOWN)، كما رأينا أعلاه. عندئذ علينا التعامل مع ثلاث قيم هى: الصحيح، والخطأ، والقيمة غير المعلومة. وفى ظل وجود القيمة غير المعلومة، علينا تفهم طريقة عمل العوامل المنطقية (AND, OR, NOT) فى ظل وجود هذه القيمة الجديدة.

إن القاعدة الرئيسية للتعامل مع هذه القيمة المنطقية الجديدة بسيطة جداً عندما نفكـر بأن قيمة الصح هى واحد (1) وأن قيمة الخطـأ هى صفر (0) وأن القيمة غير المعلومـة هـى النصف (½) (أى إنها قيمة ما بين الصحيـح والخطأ). وبناءً على ذلك يمكننا إعادة تعريف العوامل المنطقية كما يلى:

١- نتيجة العامل المنطقى «و» (AND) لأية قيمتين منطقيتين هى القيمة الدنيا للقيمتين المنطقيتين اللتين يربط بينهما، بمعنى أن (X AND Y) سـتكون خطأ (0) إذا كان أى منهمـا خطـأ (0) بغض النظر عن قيمة الآخر. أمـا إذا كانت قيمة أى منهما غير معلومـة (UNKNOWN) وليـس أى منهما خطأ، فإن النتيجة سـتكون غير معلومة (UNKNOWN). أمـا إذا كان كلاهما (صح)، فإن النتيجة سـتكون (صح). ويعنى هـذا مرة أخرى أن نـاتج العامل المنطقى «و» الذى يربط بين أى قيمتين منطقيتين هو قيمة الدنيا منهما.

٢- نتيجـة العامل المنطقى «أو» (OR) لأى قيمتين منطقيتين هى القيمة العليا للقيمتين المنطقيتين التين يربط بينهما بمعنى أن (X OR Y) ستكون (صح) (True) إذا كان أى منهما (صح)، وغيـر معلومة إذا لم يكن أى منهما (صح) ويوجد على الأقل واحدة منهمـا غير معلومة (UNKNOWN). أما إذا كان كلاهما (خطأ) فإن نتيجة العامل المنطقى (خطأ).

٣- نفـى أى قيمـة منطقية (V) هو (1-V). ويعنى هـذا أن نفى قيمة (X) عندما تكون صـح هو (١- قيمة X) وهى صفر، أى خطـأ، فى هذه الحالة. أما نفى قيمة (X) عندمـا تكون خطأ فهى (١- قيمة X) وهـى واحد، أى صح، فى هذه الحالة. أما نفى قيمة (X) عندما تكون غير معلومة فهى (١- قيمة X) وهى (½)، أى قيمة غير معلومة، فى هذه الحالة.

وتلخص الجداول الثلاثة التالية القيم المنطقية عند تطبيق العوامل المنطقية الثلاث (AND, OR and NOT).

AND	T	F	U
T	T	F	U
F	F	F	F
U	U	F	U

OR	T	F	U
T	T	T	T
F	T	F	U
U	T	U	U

NOT	
T	F
F	T
U	U

عند تطبيق شـــرط الاسترجاع (WHERE) على حقل ما ضمن تعليمة الاختيار، فإن لغة الاستفســـار البنائية تقوم بإظهار كل سجل تكون نتيجة تطبيق شرط البحث عليه مساوية للقيمة المنطقية الصحيحة فقط. ويعنى هذا أنه إذا كانت القيمة المخزنة فى الحقل المطبق عليه شـــرط الاسترجاع فى أحد السجلات مساوية للقيمة غير المعرفة أو أن نتيجة تطبيق شـــرط الاســـترجاع عليه هى القيمة (خطأ)، فإن مثل هذا السجل لـــن يظهر ضمن نتيجة تعلية الاختيار. فعلى ســـبيل المثال، لنفتـــرض أننا نرغب فى الاستفســـار عن أعضاء هيئة التدريس الذين تزيد رواتبهم عن ٤٠,٠٠٠ ألفاً. عندئذ ستكون تعليمة لغة الاستفسار البنائية المناسبة لهذا الاستفسار كما يلى:

```
SELECT *
FROM FACULTY_T
WHERE SALARY > 40000;
```

وتكون نتيجة التعليمة الســـابقة عبارة عن ســـبعة سجلات لسبعة من أعضاء هيئة التدريس تتوافق مرتباتهم مع شرط الاسترجاع، كما يلى:

```
FACULTY_ FNAME        LNAME        PHONE_NO      SALARY DOB       DEPART
-------- ------------ ------------ -------- ---------- --------- ------
320      Mohammed     Alhamad      454-5412      44000 13-MAY-65 CS
330      Ghanim       Alghanim     456-2234      44500 12-AUG-69 CS
420      Saleh        Alghamdi     454-2233      44600 13-FEB-69 CHEM
640      Fahad        Alzaid       456-3322      44300 12-MAY-71 STAT
660      Saud         Alkhalifa    454-9856      44900 13-AUG-72 STAT
770      Sultan       Aljasir      456-3212      43300 13-MAY-70 PHYS
800      Ali          Albader      456-7812      45300 22-JUN-66 EE
810      Saad         Alzhrani     454-5578      44200 17-OCT-67 EE
```

ولنفتـــرض الآن أننـــا قمنا بتعديل راتب كل من عضو هيئـــة التدريس رقمه (٣٢٠) وعضو هيئة التدريس رقمه (٨١٠) بحيث تصبح غير معرفة (NULL). عندئذ ستكون نتيجة تنفيذ التعليمة كما يلى:

```
FACULTY_ FNAME        LNAME        PHONE_NO      SALARY DOB       DEPART
-------- ------------ ------------ -------- ---------- --------- ------
330      Ghanim       Alghanim     456-2234      44500 12-AUG-69 CS
420      Saleh        Alghamdi     454-2233      44600 13-FEB-69 CHEM
640      Fahad        Alzaid       456-3322      44300 12-MAY-71 STAT
660      Saud         Alkhalifa    454-9856      44900 13-AUG-72 STAT
770      Sultan       Aljasir      456-3212      43300 13-MAY-70 PHYS
800      Ali          Albader      456-7812      45300 22-JUN-66 EE
```

ويعنى هــذا أن أعضاء هيئة التدريس الذين لا ينطبق عليهم شــرط الاســترجاع (رواتبهم ليست أكثر من ٤٠,٠٠٠ ألفاً) وأولئك الذين يحتوى حقل رواتبهم على القيمة غيــر المعرفة لــن تظهر بياناتهم ضمن عملية الاختيار. ويعنــى هذا مرة أخرى أن ما يظهــر ضمن نتائج عملية الاختيار هى تلك الســجلات التى تكون فيها نتيجة شــرط الاسترجاع صحيحة فقط. أما تلك السجلات التى تكون فيها نتيجة شرط الاسترجاع خطأ أو غير معلومة فلن تظهر من ضمن نتائج التعليمة.

أمــا إذا أردنا اســترجاع الســجلات ذات القيم غير المعرفة، فإن لغة الاستفســار البنائية تقدم معامل مقارنة خاص لهذا الغرض وهو (IS NULL)، كما يلى:

```
WHERE ColumnName IS NULL
              أو
WHERE ColumnName IS NOT NULL
```

فلو أردنا استرجاع سجلات أعضاء هيئة التدريس ذوى الرواتب غير المعرفة، يمكن استخدام تعليمة الاختيار التالية:

```
SELECT *
FROM FACULTY_T
WHERE SALARY IS NULL;
```

وتكون نتيجة التعليمة السابقة السجلات التالية:

FACULTY_	FNAME	LNAME	PHONE_NO	SALARY	DOB	DEPART
320	Mohammed	Alhamad	454-5412		13-MAY-65	CS
810	Saad	Alzhrani	454-5578		17-OCT-67	EE

أما إذا أردنا استرجاع ســجلات أعضاء هيئة التدريس الذين لديهم رواتب، يمكن استخدام التعليمة التالية:

```
SELECT *
FROM FACULTY_T
WHERE SALARY IS NOT NULL;
```

وتكون نتيجة تنفيذ التعليمة الســابقة، والتى لا يظهر من ضمنها ســجلات أعضاء هيئة التدريس ذوى الرواتب غير المعرفة، كما يلى:

```
FACULTY_ FNAME        LNAME        PHONE_NO     SALARY DOB       DEPART
-------- ------------ ------------ -------- ---------- --------- ------
200      Khalid       Aloufi       454-2341      35000 22-MAY-63 MATH
220      Fahad        Alhamid      456-7733      25900 07-OCT-70 MATH
310      Saleh        Aleesa       454-8932      30000 13-SEP-66 CS
330      Ghanim       Alghanim     456-2234      44500 12-AUG-69 CS
340      Ibraheem     Alsaleh      454-1234      25000 20-JAN-70 CS
400      Ahmad        Alotaibi     454-4563      33900 17-MAY-71 CHEM
420      Saleh        Alghamdi     454-2233      44600 13-FEB-69 CHEM
500      Yahya        Khorshid     456-2221      36700 12-MAR-65 ENGL
540      Salem        Alhamad      456-3304      40000 11-SEP-72 ENGL
560      Salman       Albassam     454-7865      33800 13-SEP-68 ENGL
600      Turki        Alturki      456-7891      27800 23-JUL-75 STAT
640      Fahad        Alzaid       456-3322      44300 12-MAY-71 STAT
660      Saud         Alkhalifa    454-9856      44900 13-AUG-72 STAT
710      Mahmood      Alsalem      456-3323      31900 19-FEB-73 PHYS
730      Mishal       Almazid      454-2343      29800 17-SEP-75 PHYS
770      Sultan       Aljasir      456-3212      43300 13-MAY-70 PHYS
800      Ali          Albader      456-7812      45300 22-JUN-66 EE
850      Ahmad        Alsabti      456-0120      33900 15-APR-73 EE
```

ويمكن استخدام عامل المقارنة (IS NULL) و (IS NOT NULL) مع أية توليفات أخرى من عوامل المقارنة المنطقية التى سبق التطرق إليها ضمن شرط الاسترجاع.

٧-٢-١-٦ ترتيب نتيجة عملية الاختيار باستخدام عبارة (ORDER BY):

على الرغم من وجود ترتيب افتراضى للحقول ضمن الجداول وهو نفس ترتيبها فى تعليمة الإنشاء الذى تم استخدامها لإنشاء الجداول، إلا أنه لا يوجد ترتيب افتراضى للسجلات فى أى جدول. لذلك توفر لغة الاستفسار البنائية عبارة الترتيب (Order By)، وذلك عند الرغبة فى ترتيب نتيجة عملية الاختيار وفق قيم حقل أو أكثر من حقول الجدول. ويمكن ترتيب قيم أى حقل ضمن نتيجة العملية إما بشكل تصاعدى وإما بشكل تنازلى. أما إذا تضمن الحقل الذى ستجرى عليه عملية الترتيب قيماً غير معرفة فإنها تظهر فى نهاية النتيجة إذا كان الترتيب تصاعدياً وفى بداية النتيجة إذا كان الترتيب تنازلياً. كما أن الحالة الافتراضية للترتيب فى حالة عدم تحديد طريقة الترتيب على قيم الحقل فهى الترتيب التصاعدي. لذلك يمكننا الاستغناء عن ذكر طريقة الترتيب فى حالة رغبتنا فى إجراء الترتيب التصاعدى على قيم حقل ما. وفيما يلى الشكل العام لعبارة الترتيب فى تعليمة الاختيار.

```
SELECT Column_Name(s)
FROM Tabel_Name
WHERE Condition
ORDER BY ColumnName [ ASC ] [, ColumnName [ DESC ] ...];
```

ومثالٌ على عبارة الترتيب، لنفترض أننا نرغب فى استرجاع سجلات أعضاء هيئة التدريـس الذين تزيد رواتبهم عن ٤٠,٠٠٠ ألفاً، وترتيب النتيجة تنازلياً وفقاً للرواتب التى يتقاضونها. فى هذه الحالة يمكن استخدام تعليمة الاختيار التالية:

```
SELECT *
FROM FACULTY_T
WHERE SALARY > 40000
ORDER BY SALARY DESC;
```

وتكون نتيجة العملية الجدول التالى الذى يلاحظ فيه ترتيب السجلات تنازلياً وفقاً للرواتب التى يتقاضاها أعضاء هيئة التدريس:

FACULTY_	FNAME	LNAME	PHONE_NO	SALARY	DOB	DEPART
800	Ali	Albader	456-7812	45300	22-JUN-66	EE
660	Saud	Alkhalifa	454-9856	44900	13-AUG-72	STAT
420	Saleh	Alghamdi	454-2233	44600	13-FEB-69	CHEM
330	Ghanim	Alghanim	456-2234	44500	12-AUG-69	CS
640	Fahad	Alzaid	456-3322	44300	12-MAY-71	STAT
770	Sultan	Aljasir	456-3212	43300	13-MAY-70	PHYS

أما إذا أردنا اســترجاع ســجلات أعضاء هيئة التدريس الذيــن تزيد رواتبهم عن ٤٠,٠٠٠ ألفاً، وأولئك ذوى الرواتب غير المعرفة، وترتيب النتيجة تنازلياً وفقاً للرواتب التى يتقاضونها، فإنه يمكن استخدام تعليمة الاختيار التالية:

```
SELECT *
FROM FACULTY_T
WHERE SALARY > 40000 OR SALARY IS NULL
ORDER BY SALARY DESC;
```

ويلاحظ فى الجدول التالى الذى يمثل نتيجة التعليمة السابقة أنه قد تم استرجاع السجلات المطلوبة حسب شرط الاسترجاع وترتيبها **تنازلياً** حسب الراتب. كما يلاحظ ظهور سجلات أعضاء هيئة التدريس ذوى الرواتب غير المعرفة فى بداية الترتيب.

FACULTY_	FNAME	LNAME	PHONE_NO	SALARY	DOB	DEPART
320	Mohammed	Alhamad	454-5412		13-MAY-65	CS
810	Saad	Alzhrani	454-5578		17-OCT-67	EE
800	Ali	Albader	456-7812	45300	22-JUN-66	EE
660	Saud	Alkhalifa	454-9856	44900	13-AUG-72	STAT
420	Saleh	Alghamdi	454-2233	44600	13-FEB-69	CHEM
330	Ghanim	Alghanim	456-2234	44500	12-AUG-69	CS
640	Fahad	Alzaid	456-3322	44300	12-MAY-71	STAT
770	Sultan	Aljasir	456-3212	43300	13-MAY-70	PHYS

أما إذا أردنا استرجاع سجلات أعضاء هيئة التدريس الذين تزيد رواتبهم عن ٤٠,٠٠٠ ألفاً وأولئك ذوى أصحاب الرواتب غير المعرفة، وترتيب النتيجة **تصاعدياً** وفقاً للرواتب التى يتقاضونها، فإنه يمكن استخدام تعليمة الاختيار التالية التى يلاحظ فيها عدم استخدام الكلمة المحجوزة (ASC)، لأن الترتيب يكون تصاعدياً بشكل افتراضى عند عدم تحديد ترتيب معين للنتيجة.

```
SELECT *
FROM FACULTY_T
WHERE SALARY > 40000 OR SALARY IS NULL
ORDER BY SALARY;
```

ويلاحظ فى الجدول التالى الذى يمثل نتيجة التعليمة السابقة أنه قد تم استرجاع السجلات المطلوبة حسب شرط الاسترجاع وترتيبها تصاعدياً حسب الراتب. كما يلاحظ ظهور سجلات أعضاء هيئة التدريس ذوى الرواتب غير المعرفة فى نهاية الترتيب.

FACULTY_	FNAME	LNAME	PHONE_NO	SALARY	DOB	DEPART
770	Sultan	Aljasir	456-3212	43300	13-MAY-70	PHYS
640	Fahad	Alzaid	456-3322	44300	12-MAY-71	STAT
330	Ghanim	Alghanin	456-2234	44500	12-AUG-69	CS
420	Saleh	Alghamdi	454-2233	44600	13-FEB-69	CHEM
660	Saud	Alkhalifa	454-9856	44900	13-AUG-72	STAT
800	Ali	Albader	456-7812	45300	22-JUN-66	EE
320	Mohammed	Alhamad	454-5412		13-MAY-65	CS
810	Saad	Alzhrani	454-5578		17-OCT-67	EE

ومن المثالين السابقين يمكن أن نستخلص أن القيمة غير المعرفة تعد العليا من حيث الترتيب بمعنى أنها تظهر فى بداية النتيجة عندما يكون الترتيب تنازلياً وفى نهاية الترتيب عندما يكون الترتيب تصاعدياً.

كما يمكن ترتيب نتيجة عملية الاختيار وفق قيم أكثر من حقل كما أسلفنا سابقاً. فعلى سبيل المثال لنفترض أننا نرغب فى استرجاع سجلات أعضاء هيئة التدريس

الذين تزيد رواتبهم على ٤٠,٠٠٠ ألفاً، وترتيب النتيجة تصاعدياً وفقاً لرموز الأقسام الدراســية التى يتبعونها، وتنازلياً وفقاً لأسماء عائلاتهم بحيث يظهر اسم القسم أولاً متبوعاً باســم عضو هيئة التدريس. وللحصول على النتيجة المطلوبة يمكن استخدام تعليمة الاختيار التالية:

```
SELECT DEPARTMENT_ID, LNAME
    FROM FACULTY_T
    WHERE SALARY > 40000
    ORDER BY DEPARTMENT_ID ASC,  LNAME DESC;
```

ويلاحظ فى الجدول التالى الذى يمثل نتيجة التعليمة السابقة أنه قد تم استرجاع الســجلات المطلوبة حسب شرط الاســترجاع (بحيث يكون الراتب أكثر من ٤٠,٠٠٠ ألفاً) وترتيبها تصاعدياً حســب اســم القسم الدراســى (أولاً)، ومن ثم ترتيب أسماء أعضاء هيئة التدريس تنازلياً حسب أسماء عائلاتهم، حيث يلاحظ أن قسم الإحصاء قد جاء فى نهاية الترتيب التصاعدى وفقاً لترتيب الأقسام الدراسية، وأن عضو هيئة التدريس ذا اســم العائلة «الزيد» (Alzaid) قد جاء قبل عضو هيئة التدريس ذى اســم العائلة «الخليفة» (Alkhalifa) بشكل تنازلى داخل رمز قسم الإحصاء.

```
DEPART LNAME
------ ------------
CHEM   Alghamdi
CS     Alghanim
EE     Albader
PHYS   Aljasir
STAT   Alzaid
STAT   Alkhalifa
```

٧-٢-١-٦-١ ترتيب النتائج وفقاً للأرقام النسبية للأعمدة:

تمكن لغة الاستفســار البنائية من ترتيب نتائج عمليــة الاختيار، فى عبارة ترتيب نتائج عملية الاختيار (ORDER BY)، وفقاً للأرقام النســبية للحقول التى يتم اختيارها ضمــن نتيجة عملية الاختيار عوضاً عن اســتخدام أســمائها الفعلية. ويكون شــكل التعليمة فى هذه الحالة كما يلى:

```
SELECT Column_Name(s)
FROM Tabel_Name
WHERE Condition
ORDER BY ColumnNumber1 [ ASC ] [, ColumnNumber2 [ DESC ] ...];
```

فعلى سبيل المثال، يمكن استخدام الأرقام النسبية للحقول عند الرغبة فى إظهار عناوين المواد الدراسية التى تنفذ من قبل قسم الكيمياء وقسم الحاسب الآلى مرتبة حسب رمز القسم (أولاً) بشكل تصاعدى وحسب عناوين (أو مسميات) المواد الدراسية التى ينفذها كل قسم (ثانياً) بشكل تنازلى، كما يلى:

```
SELECT DEPARTMENT_ID, TITLE
FROM COURSE_T
WHERE DEPARTMENT_ID IN ('CS', 'CHEM')
ORDER BY 1 ASC, 2 DESC;
```

وتكون نتيجة التعليمة كما يلى:

```
DEPART TITLE
------ ----------------------------------
CHEM   CHEMISTRY (II)
CHEM   CHEMISTRY (I)
CS     SOFTWARE ENGINEERING
CS     JAVA PROGRAMMING
CS     INTRODUCTION TO DATABASE SYSTEMS
CS     COMPUTER ARCHITECTURE
CS     C/C++ PROGRAMMING
```

كما يمكن استخدام الأرقام النسبية للحقول حتى لو تم اختيار جميع حقول جدول ما إذا ما عرف ترتيب الحقول وفق تعليمة إنشاء الجدول. فعلى سبيل المثال، لو أردنا إظهار أسماء أعضاء هيئة التدريس الذين يعملون فى قسم الكيمياء وقسم الحاسب الآلى مرتبة تصاعدياً حسب أسمائهم الأولى وتنازلياً حسب رواتبهم، يمكن استخدام تعليمة الاختيار وفق الصيغة التالية:

```
SELECT *
FROM FACULTY_T
WHERE DEPARTMENT_ID IN ('CS', 'CHEM')
ORDER BY 2 ASC, 5 DESC;
```

وتكون نتيجة التعليمة كما يلى:

```
FACULTY_ FNAME        LNAME        PHONE_NO     SALARY DOB       DEPART
-------- ------------ ------------ -------- ---------- --------- ------
400      Ahmad        Alotaibi     454-4563      33900 17-MAY-71 CHEM
330      Ghanim       Alghanim     456-2234      44500 12-AUG-69 CS
340      Ibraheem     Alsaleh      454-1234      25000 20-JAN-70 CS
320      Mohammed     Alhamad      454-5412      44000 13-MAY-65 CS
420      Saleh        Alghamdi     454-2233      44600 13-FEB-69 CHEM
310      Saleh        Aleesa       454-8932      30000 13-SEP-66 CS
```

ويلاحظ فى نتيجة التعليمة السابقة أنه قد تم ترتيب النتيجة وفق الأسماء الأولى بشكل تصاعدي، ومن ثم يكون ترتيب النتيجة وفقاً للراتب بشكل تنازلى. وعند تطابق الأســماء الأولى، كما فى حالة الاسم «صالح» (Saleh)، يتم الترتيب وفق الراتب بشكل تنازلى، كما توضح نتيجة التعليمة السابقة.

وتكمن أهمية ترتيب الحقول وفقاً لأرقامها النســبية بشــكل خاص عند استخدام القيــم المحســوبة التى لا تكــون عادة من ضمن حقــول الجدول، وإنما يتم حســابها باســتخدام تعليمــة الاختيار وفق حقل أو أكثر من حقول الجــدول، كما يوضح الجزء التالى من هذا الفصل.

٧-٢-١-٧ القيم المحسوبة (Computed Values):

يمكن اســتخدام التعبيرات الحاســبية ضمن تعليمة الاختيار وتطبيقها على حقول الجدول أو اســتخدام الدوال الحســابية التى توفرها لغة الاستفســار البنائية، حيث يمكــن أن تحتوى نتيجة تعليمــة الاختيار، بالإضافة لما يتم اختياره من حقول الجدول، على تعبيرات حســابية تحتوى على أســماء حقول وقيم عددية ثابتة تربطها العوامل الحســابية وهى: الضرب (*)، والقســمة (/)، والجمع (+)، والطرح (-). وكما هو الحال فى لغات برمجة الحاسب الآلي، يتم تقييم عملية الضرب وعملية القسمة فى التعبير الحسابى أولاً، ومن الجهة اليسرى للجهة اليمنى، ثم تقييم عملية الجمع وعملية الطرح ثانياً، ومن الجهة اليســرى للجهة اليمنى أيضاً. ولتغيير أولويات التقييم هذه أو إزالة الالتباس منها، يمكن استخدام الأقواس بحيث يتم تقييمها من الداخل للخارج. وعند الرغبة فى ترتيب نتائج عملية الاختيار وفق قيم محســوبة، يســتخدم الرقم النســبى لحقل القيمة المحسوبة حسب ترتيبه ضمن عبارة الاختيار (SELECT).

وثمــة مثال على القيم المحســوبة وترتيب النتائج وفقاً لهــا، لنفترض أننا نرغب فى إظهار أســماء أعضاء هيئة التدريس الذين يعملون فى قســم الكيمياء وقســم الحاسب الآلى، ورواتبهم بعد زيادتها بنســبة خمســة عشــر فى المائة (١٥٪) مرتبة تنازلياً حسب الراتب بعد زيادته. فى هذه الحالة يمكن استخدام تعليمة الاختيار التالية:

```
SELECT FNAME, LNAME, SALARY*1.15
FROM FACULTY_T
WHERE DEPARTMENT_ID IN ('CS', 'CHEM')
ORDER BY 3 DESC;
```

عند تنفيذ التعليمة السابقة، تكون نتيجتها كما يلى:

```
FNAME        LNAME        SALARY*1.15
------------ ------------ -----------
Saleh        Alghamdi           51290
Ghanim       Alghanim           51175
Mohammed     Alhamad            50600
Ahmad        Alotaibi           38985
Saleh        Aleesa             34500
Ibraheem     Alsaleh            28750
```

كما يمكن اســتخدام القيم المحســوبة ضمن عبارة شرط الاختيار (WHERE). فلو افترضنا أننا نرغب فى إظهار أســماء أعضاء هيئة التدريس الذين يعملون فى قســم الكيمياء وقسم الحاسب الآلى، وستزيد رواتبهم بعد نسبة الزيادة (١٥٪) على خمسين ألفاً، يمكن استخدام تعليمة الاختيار التالية:

```
SELECT FNAME, LNAME
FROM FACULTY_T
WHERE DEPARTMENT_ID IN ('CS', 'CHEM')
      AND SALARY*1.15 > 50000;
```

عند تنفيذ التعليمة السابقة، تكون نتيجتها كما يلى:

```
FNAME        LNAME
------------ ------------
Mohammed     Alhamad
Ghanim       Alghanim
Saleh        Alghamdi
```

وبمقارنة نتيجة هذه التعليمة بنتيجة التعليمة الســابقة يلاحظ أن الأسماء الأولى الثلاث من نتيجة التعليمة السابقة هى التى ظهرت ضمن نتيجة التعليمة الحالية لكون أســماء أعضاء هيئة التدريس هذه هى من أســماء العاملين فى قسم الكيمياء وقسم الحاســب الآلى، وســتزيد رواتبها على ٥٠,٠٠٠ ألفاً بعد زيادة الرواتب بنسبة خمسة عشر فى المائة.

كما تجدر الإشارة هنا إلى أن القيم المحسوبة عند استخدامها ضمن عملية الاختيار لا تؤثــر بأى شــكل كان فيما هو مخزن فى حقول الجــداول، خاصة إذا ما تذكرنا أن تعليمة الاختيار ما هى إلا عملية استفســار (أو عملية استرجاع) لا تؤثر فى محتويات قاعدة البيانات.

٧-٢-١-٨ دوال التجميع (أو الأعمدة) (Aggregate (or Column) Functions):

توفر لغة الاستفسار البنائية عدداً من الدوال الإحصائية المسماة «دوال التجميع» (Aggregate Functions). وتسمى هذه الدوال أحياناً «دوال الأعمدة» (Column Functions)، وذلك لكونها تطبق على حقول الجداول كأعمدة كاملة فيها. وهذه الدوال هى: دالة «العدد» (COUNT)، ودالة «القيمة الصغرى» (MIN)، ودالة «القيمة الكبرى» (MAX)، ودالة «الجمع» (SUM)، ودالة «المتوسط» (AVG). وتكون نتيجة أية دالة تجميع من دوال التجميع عبارة عن صف واحد يمثل نتيجة قيمة الدالة عوضاً عن مجموعة من الصفوف تمثل محتويات الجدول الذى تم تطبيق الدالة عليه.

تستخدم دالة العدد (COUNT) لإيجاد عدد صفوف جدول ما، ينطبق عليها شرط الاختيار (WHERE) في حالة وجوده، فى تعليمة الاختيار. فلمعرفة عدد أعضاء هيئة التدريس فى الجامعة الأهلية، يمكن استخدام تعليمة الاختيار التالية:

```
SELECT COUNT(*)
FROM FACULTY_T;
```

وتكون نتيجة التعليمة السابقة هى العدد عشرون (٢٠)، وذلك لكون عدد أعضاء هيئة التدريس فى الجامعة عشرين عضواً، كما يلى:

```
COUNT(*)
----------
        20
```

أما إذا أردنا معرفة عدد أعضاء هيئة التدريس فى كل من قسم الكيمياء وقسم الحاسب الآلى، يمكن استخدام التعليمة التالية التى تتضمن شرط الاختيار المناسب (وهو أن يكون القسم الذى يتبعه عضو هيئة التدريس، إما قسم الكيمياء وإما قسم الحاسب الآلى)، كما يلى:

```
SELECT COUNT(*)
FROM FACULTY_T
WHERE DEPARTMENT_ID IN ('CS', 'CHEM');
```

وتكون نتيجة التعليمة السابقة كما يلى:

```
COUNT(*)
----------
         6
```

كما يمكن أن تستخدم دالة العدد لإيجاد عدد صفوف نتيجة عملية الاختيار مع استبعاد الصفوف المتكررة فى قيمة الحقل الذى تطبق عليه دالة العدد، وكذلك القيم غير المعرفة. فعلى سبيل المثال، يمكن معرفة عدد رواتب أعضاء هيئة التدريس المختلفة (أى دون تكرار) باستخدام التعليمة التالية:

```
SELECT COUNT(DISTINCT SALARY)
FROM FACULTY_T;
```

وتكون نتيجة التعليمة كما يلى:

```
COUNT(DISTINCTSALARY)
---------------------
                   19
```

وعلى الرغم من وجود عشرين عضواً من أعضاء هيئة التدريس فى الجامعة الأهلية، إلا أن نتيجة دالة العدد لم تظهر سوى تسعة عشر (١٩)، وذلك لكون اثنين من أعضاء هيئة التدريس يتقاضون الراتب نفسه (وهو ٣٣,٩٠٠). لذلك فإن دالة العدد التى طبقت على حقل الراتب قامت بعَدٍّ واحدٍ من هذين الراتبين فقط مع حذف المتكرر منها.

أما إذا استخدمنا دالة العدد وفق التعليمة التالية، فإنها لن تقوم بعَدٍّ أعضاء هيئة التدريس الذين لم تحدد رواتبهم بعد (أى فى حالة كون قيم رواتبهم غير معرفة (NULL)) ضمن نتيجة الدالة.

```
SELECT COUNT(SALARY)
FROM FACULTY_T;
```

أما دالة القيمة الصغرى (MIN) فتقوم بتحديد القيمة الصغرى للحقل المطبقة عليه من ضمن الصفوف التى ينطبق عليها شرط الاختيار، ودالة القيمة الكبرى (MAX) تقوم بتحديد القيمة الكبرى للحقل المطبقة عليه من ضمن الصفوف التى ينطبق عليها شرط الاسترجاع. فعلى سبيل المثال، يمكننا معرفة أصغر راتب وأكبر راتب يتقاضاه أعضاء هيئة التدريس العاملين فى قسم الكيمياء وقسم الحاسب الآلى باستخدام التعليمة التالية:

```
SELECT MIN(SALARY), MAX(SALARY)
FROM FACULTY_T
WHERE DEPARTMENT_ID IN ('CS', 'CHEM');
```

وتكـــون نتيجة التعليمة الســـابقة عبارة عن ٢٥,٠٠٠ للقيمـــة الصغرى (أقل راتب يتقاضـــاه أعضاء هيئة التدريس فى كلا القســـمين)، و ٤٤,٦٠٠ للقيمة الكبرى (أعلى راتب يتقاضاه أعضاء هيئة التدريس فى كلا القسمين) وكما يلى:

```
MIN(SALARY) MAX(SALARY)
----------- -----------
      25000       44600
```

تجدر الإشـــارة إلى أنه فى حالة وجود قيم غير معرفة فى الحقل الذى تطبق عليه القيمة الصغـــرى أو القيمة الكبرى، فإن هذه القيم يتم تجاهلها فى كلتا العمليتين ولا تدخل ضمن نتيجتيهما. كما تجدر الإشـــارة إلى أنه بالإمكان استخدام كلتا العمليتين مع حقول ليســـت من نوع الأعداد حيث يمكن اســـتخدامهما مع السلاســـل الحرفية، والتاريـــخ والوقت، كذلك. فمثـــلاً، يمكننا التعرف على أصغر اســـم عائلة (من حيث الترتيب الأبجدى) وأكبر اسم عائلة لأعضاء هيئة التدريس العاملين فى قسم الكيمياء وقسم الحاسب الآلى وفق التعليمة التالية:

```
SELECT MIN(LNAME), MAX(LNAME)
FROM FACULTY_T
WHERE DEPARTMENT_ID IN ('CS', 'CHEM');
```

وتكون نتيجة التعليمة أن اســـم العائلة (Aleesa) هو الأصغر أبجدياً واســـم العائلة (Alsaleh) هو الأكبر، كما يلى:

```
MIN(LNAME)   MAX(LNAME)
------------ ------------
Aleesa       Alsaleh
```

أما دالة المتوســـط (AVG) فتستخدم لحساب القيمة المتوسطة للحقل المطبقة عليه فى الصفوف التى ينطبق عليها شـــرط الاختيار مع تجاهل القيم غير المعرفة فى حال وجودها. فعلى ســـبيل المثال، يمكن حســـاب متوســـط رواتب أعضاء هيئة التدريس العاملين فى الجامعة الأهلية باستخدام التعليمة التالية:

```
SELECT AVG(SALARY)
FROM FACULTY_T;
```

وتكون نتيجة التعليمة كما يلى:

```
AVG(SALARY)
-----------
      36940
```

أما إذا أردنا حساب متوسط رواتب أعضاء هيئة التدريس دون اعتبار المتكرر منها فإنه يتم اســتخدام عبارة (DISTINCT). وفى هذه الحالة يتم إدخال أى قيمة متكررة مرة واحدة فقط عند حساب المتوسط، كما يلى:

```
SELECT AVG(DISTINCT SALARY)
FROM FACULTY_T;
```

ونظراً لتكرار الراتب ٣٣,٩٠٠ مرتين، تكون نتيجة العملية السابقة كما يلى:

```
AVG(DISTINCTSALARY)
-------------------
              37100
```

كما يمكن استخدام دالة المتوسط على مجموعة جزئية من صفوف الجدول ينطبق عليها شــرط الاختيار. فمثلاً، يمكن حســاب متوســط رواتب أعضاء هيئة التدريس العاملين فى قسم الكيمياء وقسم الحاسب الآلى وفق التعليمة التالية:

```
SELECT AVG(SALARY)
FROM FACULTY_T
WHERE DEPARTMENT_ID IN ('CS', 'CHEM');
```

وتكون نتيجة التعليمة السابقة كما يلى:

```
AVG(SALARY)
-----------
      37000
```

أما دالة الجمع (SUM) فتستخدم لجمع قيم الحقل المطبقة عليه فى الصفوف التى ينطبــق عليها شــرط الاختيار مع تجاهل القيم غير المعرفــة فى حال وجودها. فعلى ســبيل المثال، يمكن حساب مجموع رواتب أعضاء هيئة التدريس العاملين فى الجامعة الأهلية باستخدام التعليمة التالية:

```
SELECT SUM(SALARY)
FROM FACULTY_T;
```

وتكون نتيجة التعليمة كما يلى:

```
SUM(SALARY)
-----------
     738800
```

أمـــا إذا أردنا معرفة مجموع رواتب أعضاء هيئة التدريس دون اعتبار المتكرر منها، فإنه يتم اســـتخدام عبارة (DISTINCT). وفى هذه الحالة يتم إدخال أية قيمة متكررة مرة واحدة فقط عند حساب المجموع، كما يلى:

```
SELECT SUM(DISTINCT SALARY)
FROM FACULTY_T;
```

ونظراً لتكرار الراتب ٣٣,٩٠٠ مرتين، تكون نتيجة العملية السابقة كما يلى:

```
SUM(DISTINCTSALARY)
-------------------
             704900
```

كمـــا يمكن اســـتخدام دالة الجمع على مجموعة جزئية مـــن صفوف الجدول التى ينطبق عليها شرط الاختيار. فمثلاً، يمكن حساب مجموع رواتب أعضاء هيئة التدريس العاملين فى قسم الكيمياء وقسم الحاسب الآلى وفق التعليمة التالية:

```
SELECT SUM(SALARY)
FROM FACULTY_T
WHERE DEPARTMENT_ID IN ('CS', 'CHEM');
```

وتكون نتيجة التعليمة السابقة كما يلى:

```
SUM(SALARY)
-----------
     222000
```

٧-٢-١-٩ عبارة التجميع (GROUP BY):

فـــى الكثير مـــن الأحيان تظهر الحاجة إلى تطبيـــق دوال التجميع على مجموعات جزئية من ســـجلات جدول ما. فعلى ســـبيل المثال، قد نحتاج إلى معرفة متوســـط، أو مجمـــوع رواتب أعضاء هيئة التدريس فى كل قســـم دراســـى علـــى حدة. فى هذه الحالة نحتاج إلى تقسيم سجلات الجدول إلى مجموعات غير متداخلة من السجلات

بحيث تحتوى كل مجموعة على السـجلات التـى تتوافق مع بعضها فى الحقول المراد التقسيم عليها. وتسمى هذه الحقول «حقول التقسيم» أو «حقول التجميع» (Grouping Attributes). بعد ذلك نقوم باسـتخدام دالة التجميع المراد تطبيقها على كل مجموعة بشكل مسـتقل. وتوفر لغة الاستفسار البنائية عبارة «التجميع حسب» (GROUP BY) لهذا الغرض. وتحدد عبارة «التجميع حسب» الحقول التى سيتم تقسيم الجدول وفقاً لها، والتى **يجب** أن تكون من ضمن الحقول التى ستظهر ضمن نتيجة عبارة الاختيار. وبهذا الشكل سوف تظهر قيمة الحقول التى تم تقسيم الجدول وفقاً لها مع قيمة دالة التجميع التى تم تطبيقها على الجدول.

فعلى سـبيل المثال، يمكن معرفة عـدد أعضاء هيئة التدريس، ومتوسـط رواتبهم، ومجموع رواتبهم حسب الأقسام الدراسية التى يعملون فيها باستخدام التعليمة التالية:

```
SELECT DEPARTMENT_ID, COUNT(*), AVG(SALARY), SUM(SALARY)
FROM FACULTY_T
GROUP BY DEPARTMENT_ID;
```

يلاحظ فى التعليمة السـابقة وجود رمز القسم الدراسى ضمن الحقول التى تمثل نتيجة عبارة الاختيار التى يتم تقسيم الجدول وفقاً لقيمته فى عبارة «تجميع حسب»، الأمر الذى يعد ضرورياً لعمل عبارة «تجميع حسب». بالإضافة إلى ذلك يظهر ضمن عبارة الاختيار دوال التجميع الواجب تطبيقها على كل مجموعة من المجموعات الناتجة من عملية تقسـيم الجدول. وتكون نتيجة التعليمة السـابقة، التى تظهر عدد أعضاء هيئة التدريس فى كل قسم دراسى، ومتوسط رواتبهم حسب القسم الذى يعملون فيه، ومجموع رواتب أعضاء هيئة التدريس فى كل قسم دراسى، كما يلى:

```
DEPART   COUNT(*) AVG(SALARY) SUM(SALARY)
------ ---------- ----------- -----------
CHEM            2       39250       78500
CS              4       35875      143500
EE              3  41133.3333      123400
ENGL            3  36833.3333      110500
MATH            2       30450       60900
PHYS            3       35000      105000
STAT            3       39000      117000
```

وإذا أردنا ترتيب نتائج العملية السابقة أبجدياً وبشكل تنازلى حسب رموز الأقسام الدراسية، فإنه يمكن استخدام عبارة «ترتيب حسب» (ORDER BY)، كما يلى:

```
SELECT DEPARTMENT_ID, COUNT(*), AVG(SALARY), SUM(SALARY)
FROM FACULTY_T
GROUP BY DEPARTMENT_ID
ORDER BY DEPARTMENT_ID DESC;
```

وتكون نتيجة التعليمة السابقة كما يلى:

```
DEPART   COUNT(*)  AVG(SALARY)  SUM(SALARY)
------ ---------- ----------- -----------
STAT            3       39000      117000
PHYS            3       35000      105000
MATH            2       30450       60900
ENGL            3  36833.3333      110500
EE              3  41133.3333      123400
CS              4       35875      143500
CHEM            2       39250       78500
```

أمـــا إذا أردنا معرفة عدد المواد الدراســية المســجل فيهــا كل طالب فى كل فصل دراسى، وحسب السنة الدراســية، وترتيب النتيجة تصاعدياً حسب الأرقام الدراسية للطلبة، نستخدم التعليمة التالية:

```
SELECT STUDENT_ID, YEAR, SEMESTER, COUNT(*)
FROM ENROLLMENT_T
GROUP BY STUDENT_ID, YEAR, SEMESTER
ORDER BY STUDENT_ID;
```

ويلاحظ فى النتيجة التالية للتعليمة أن الطالب رقم ١٩٩٩٢٠٢٠ قد قام بتسجيل أربع مواد دراســية فى فصل الخريف من عام ٢٠٠٠، وأربع مواد دراســية فى فصل الربيع من عــام ٢٠٠٠ أيضاً، على حين قام الطالب رقم ١٩٩٩٢٣٤١ بتســجيل ثلاث مواد فى فصل الخريف من عام ٢٠٠٠، ولم يسجل أية مادة أخرى ضمن أى فصل دراسى آخر.

```
STUDENT_        YEAR SEMESTER     COUNT(*)
-------- ---------- ---------- ----------
19992020        2000 FALL                4
19992020        2000 SPRING              4
19992341        2000 FALL                3
19994512        2000 FALL                3
20001111        2000 FALL                4
20001111        2000 SPRING              4
20001212        2000 FALL                1
```

وفى حالة وجود قيم غير معرفة ضمن أحد حقول التجميع، فإنه يتم إنشاء مجموعة مستقلة للسجلات التى تكون قيم حقول التجميع فيها غير معرفة.

٧-٢-١-١٠ عبارة ترشيح المجموعات الفرعية (HAVING):

تستخدم عبارة (HAVING) بشكل اختيارى عند وجود عبارة «تجميع حسب»، وذلك لترشيح المجموعات الفرعية بحيث يجب أن تتحقق شروط عبارة (HAVING) على المجموعات الفرعية حتى يمكن أن تظهر ضمن نتيجة تعليمة الاختيار. ويعنى هذا أن عبارة (HAVING) تحتوى على شروط يجب أن تحقق على المجموعات الفرعية الناتجة من عبارة «التجميع حسب» حتى يمكن أن تظهر ضمن تعليمة الاختيار. ويمكن تشبيه عبارة (HAVING) بعبارة (WHERE) حيث إن كلتيهما تمثلان شروطاً لعملية الاختيار، إلا أن (WHERE) تحتوى على شروط لاختيار السجلات من الجدول، فى حين أن (HAVING) تحتوى على شروط لاختيار المجموعات الفرعية.

فعلى سبيل المثال، إذا أردنا معرفة أرقام الطلاب الذين قاموا بالتسجيل فى مواد دراسية يقل عددها عن أربع مواد فى أى فصل دراسى من أى عام دراسي، وترتيب النتيجة تصاعدياً حسب الأرقام الدراسية للطلبة، نستخدم التعليمة التالية:

```
SELECT STUDENT_ID, YEAR, SEMESTER, COUNT(*)
FROM ENROLLMENT_T
GROUP BY STUDENT_ID, YEAR, SEMESTER
HAVING COUNT(*) < 4
ORDER BY STUDENT_ID;
```

وتكون نتيجة التعليمة السابقة كما يلى:

```
STUDENT_        YEAR SEMESTER     COUNT(*)
-------- ---------- ---------- ----------
19992341        2000 FALL               3
19994512        2000 FALL               3
20001212        2000 FALL               1
```

ويلاحظ فى النتيجة السابقة أنه قد تم تقسيم سجلات الجدول إلى مجموعات حسب رمز الطالب، والسنة الدراسية، والفصل الدراسى. بعد ذلك تم حساب عدد المواد الدراسية التى تم التسجيل فيها حسب هذا التقسيم وتطبيق شروط عبارة (HAVING) على المجموعات الفرعية، ومن ثم إظهار المجموعات الفرعية التى تنطبق عليها شروط عبارة (HAVING).

ويمكن تصور أولويات تنفيذ التعليمة السـابقة من قبل نظام إدارة قاعدة البيانات كما يلى:

١- اختيار الجدول المناسب حسب ذكره فى عبارة مصدر الاختيار (FROM).

٢- اختيار سجلات الجدول التى تحقق شروط عبارة شرط الاختيار (WHERE).

٣- تقسـيم الصفوف التى تحقق شـرط الاختيار إلى مجموعات فرعية حسب حقول التقسيم فى عبارة (GROUP BY).

٤- حذف المجموعات التى لا تحقق الشروط الواردة فى عبارة (HAVING).

٥- تنفيـذ دوال التجميع والتعبيرات الحسـابية الواردة فى عبارة الاختيار (SELECT) على المجموعات الفرعية التى حققت شروط عبارة (HAVING) أعلاه.

٦- ترتيب نتائج عبارة الاختيار (SELECT) الناتجة من الخطوة السـابقة حسب عبارة الترتيب (OREDRE BY).

٧-٢-١-١١ استخدام تعليمات المجموعات لدمج نتائج تعليمات اختيار متعددة:

حيـث إن نتيجة أى عمليـة اختيار عبارة عن مجموعة متعـددة (Multiset or Bag)، لكونهـا قد تحتوى على سـجلات متكررة على خلاف المجموعـات التى لا تكرر فيها العناصر، فإنه من الطبيعى أن توفر لغة الاستفسار البنائية ثلاث تعليمات للتعامل مع نتائج الاستفسـارات على أنها مجموعـات. وهذه التعليمات هى: الاتحاد (UNION)، والتقاطع (INTERSECT)، والفرق (MINUS).

٧-٢-١-١١-١ الاتحاد (UNION):

تسـتخدم تعليمة الاتحاد (UNION) لدمج نتائج تعليمات اختيار متعددة فى جدول نتائج واحد. وعلى الرغم من أن نتائج عمليات الاختيار قد تكون من جداول مختلفة، إلا أن هذه النتائج يجب أن تكون متوافقة من حيث عملية الاتحاد، بمعنى أنها يجب أن تكون بالعدد نفسه من الحقول ومن النوعية نفسها من البيانات (كما سبق أن أوضحنا فى الجزء ٤-٢-٢-١-١). ومن القواعد المتبعة فى عملية الاتحاد ما يلى:

١- تنفذ تعليمات الاختيار بالتسلسل.

٢- يجب أن تحتوى جميع عمليات الاختيار التى تربطها عملية الاتحاد على العدد نفسه من الحقول ومن النوعية نفسها من البيانات.

٣- تكون نتيجة عملية الاتحاد جدولاً واحداً لا تكرر فيه قيم السجلات.

٤- يمكن ترتيب نتيجة عملية الاتحاد باستخدام عبارة الترتيب (ORDER BY) التى تكون بعد آخر عملية اختيار مع استخدام الأرقام النسبية للحقول التى سيتم ترتيب النتائج وفقاً لها (فى حالة عدم توافق مسمياتها).

٥- عند الرغبة فى إظهار السجلات المتكررة، يمكن استخدام عبارة (UNION ALL) عوضاً عن عبارة (UNION) فقط.

فعلى سبيل المثال، لنفترض أننا نرغب فى معرفة أرقام أعضاء هيئة التدريس المؤهلين لتدريس أكثر من مادتين دراسيتين أو أعضاء هيئة التدريس الذين تقل رواتبهم أو تساوى ٣٥،٠٠٠ ألفاً، وترتيب النتيجة تصاعدياً وفق أرقام أعضاء هيئة التدريس. فى مثل هذه الحالة يمكن استخدام تعليمة الاتحاد التالية التى تربط بين نتائج تعليمتى اختيار:

```
SELECT FACULTY_ID
FROM QUALIFICATION_T
GROUP BY FACULTY_ID
HAVING COUNT(*) > 2
UNION
SELECT FACULTY_ID
FROM FACULTY_T
WHERE SALARY <= 35000
ORDER BY FACULTY_ID;
```

وتكون نتيجة تعليمة الاختيار الأولى التى تظهر أرقام هيئة التدريس المؤهلين لتدريس أكثر من مادتين دراسيتين كما يلى:

```
FACULTY_
--------
200
220
```

أما نتيجة تعليمة الاختيار الثانية التى تظهر أرقام هيئة التدريس المؤهلين الذين تقل رواتبهم أو تساوى ٣٥،٠٠٠ ألفاً فهى كما يلى:

```
FACULTY_
--------
200
220
310
340
400
560
600
710
730
850
```

ونظـــراً لكون نتائج كلتا عمليتى الاختيـــار متوافقة من حيث الاتحاد، تحتويان على العدد نفســـه من الحقول ومن النوعية نفسها من البيانات، فإنه يمكن استخدام عملية الاتحاد بينهما ومن ثم ترتيب النتيجة تصاعدياً وفقاً لأرقام أعضاء هيئة التدريس، كما توضح النتيجة النهائية للتعليمة بشـــكلها النهائى الذى تم فيه إلغاء الصفوف المتكررة من النتيجة النهائية (وإظهارها مرة واحدة فقط)، كما يلى:

```
FACULTY_
--------
200
220
310
340
400
560
600
710
730
850
```

أما إذا أردنا إظهار الســـجلات فى حال تكرارهـــا ضمن نتيجتى عمليتى الاختيار، فإنه يمكن استخدام تعليمة الاتحاد وفق الصيغة التالية:

```
SELECT FACULTY_ID
FROM QUALIFICATION_T
GROUP BY FACULTY_ID
HAVING COUNT(*) > 2
UNION ALL
SELECT FACULTY_ID
FROM FACULTY_T
WHERE SALARY <= 35000
ORDER BY FACULTY_ID;
```

وتكـــون نتيجـــة التعليمة التى يظهر فيهـــا رقم عضو هيئة التدريـــس (٢٠٠) ورقم عضو هيئة التدريس (٢٢٠) بشـــكل متكرر (لكونهما مؤهلين لتدريس أكثر من مادتين دراسيتين وتقل رواتبهما أو تساوى ٣٥,٠٠٠) كما يلى:

```
FACULTY_
--------
200
200
220
220
310
340
400
560
600
710
730
850
```

٧-٢-١-١١-٢ التقاطع (INTERSECT):

تســـتخدم تعليمة التقاطع (INTERSECT) لإظهار الســـجلات المشتركة الناتجة من تعليمـــات اختيار متعـــددة فى جدول نتائج واحد. وعلى الرغـــم من أن نتائج عمليات الاختيـــار قد تكون من جداول مختلفـــة، إلا أن هذه النتائج يجب أن تكون متوافقة من حيـــث عملية الاتحاد (كما ســـبق أن أوضحنا فى الجزء ٤-٢-٢-١-١)، كما هو الحال بالنســـبة لتعليمة الاتحاد التى ســـبق شـــرحها أعلاه. ومن القواعد المتبعة فى عملية التقاطع ما يلى:

١- تنفذ تعليمات الاختيار بالتسلسل.

٢- يجـــب أن تحتوى جميع عمليات الاختيار التى تربطهـــا عملية التقاطع على العدد نفسه من الحقول ومن النوعية نفسها من البيانات.

٣- تكون نتيجة عملية التقاطع جدولاً واحداً لا تكرر فيه قيم السجلات.

٤- يمكن ترتيب نتيجة عملية التقاطع باســـتخدام عبـــارة الترتيب (ORDER BY) التى تكون بعد آخر عملية اختيار مع استخدام الأرقام النسبية للحقول التى سيتم ترتيب النتائج وفقاً لها (فى حالة عدم توافق مسمياتها).

فعلى سـبيل المثال، لنفتـرض أننا نرغب فى معرفة أرقام أعضـاء هيئة التدريس المؤهلين لتدريس أكثر من مادتين دراسـيتين وتقل رواتبهم أو تسـاوى ٣٥,٠٠٠ ألفاً، وترتيب النتيجة تصاعدياً وفق أرقام أعضاء هيئة التدريس. فى مثل هذه الحالة يمكن استخدام تعليمة التقاطع التالية التى تربط بين نتائج تعليمتى اختيار:

```
SELECT FACULTY_ID
FROM QUALIFICATION_T
GROUP BY FACULTY_ID
HAVING COUNT(*) > 2
INTERSECT
SELECT FACULTY_ID
FROM FACULTY_T
WHERE SALARY <= 35000
ORDER BY FACULTY_ID;
```

وتتكـون نتيجـة تعليمة التقاطع السـابقة من رقمين يمثلان أرقـام أعضاء هيئة التدريس المؤهلين لتدريس أكثر من مادتين دراسـيتين ويقل راتب كل منهم أو يسـاوى ٣٥,٠٠٠ ، كما يلى:

```
FACULTY_
--------
200
220
```

٧-٢-١-١١-٣ **الفرق (MINUS):**

تسـتخدم تعليمة الفرق (MINUS) لإظهار السـجلات الناتجة من تعليمة الاختيار الأولـى وغير موجـودة فى أى من تعليمـات الاختيار اللاحقة. وعلـى الرغم من أن نتائج عمليات الاختيار قد تكون من جداول مختلفة، إلا أن هذه النتائج يجب أن تكون متوافقـة مـن حيث عملية الاتحاد، كما هو الحال بالنسـبة لتعليمـة الاتحاد وتعليمة التقاطع اللتين سبق شرحهما أعلاه. كما تجدر الإشارة إلى أن لغة الاستفسار البنائية تستخدم مسمى «عدا» (EXCEPT) عوضاً عن مسمى «الفرق» (MINUS)، إلا أن غالبية النظم التجارية ما زالت تسـتخدم كلمة الفرق. ومن القواعد المتبعة فى عملية الفرق ما يلى:

١- تنفذ تعليمات الاختيار بالتسلسل.

٢- يجب أن تحتوى جميع عمليات الاختيار التى تربطها عملية الفرق على العدد نفسه من الحقول ومن النوعية نفسها من البيانات.

٣- تكون نتيجة عملية الفرق جدولاً واحداً لا تكرر فيه قيم السجلات.

٤- يمكن ترتيب نتيجة عملية الفرق باستخدام عبارة الترتيب (ORDER BY) التى تكون بعد آخر عملية اختيار مع استخدام الأرقام النسبية للحقول التى سيتم ترتيب النتائج وفقاً لها (فى حالة عدم توافق مسمياتها).

فعلى سبيل المثال، لنفترض أننا نرغب فى معرفة أرقام أعضاء هيئة التدريس الذين يتقاضون أكثر من ٣٥,٠٠٠ ألفاً ومؤهلين لتدريس مواد دراسية يقل عددها عن مادتين دراسيتين، وترتيب النتيجة تصاعدياً وفق أرقام أعضاء هيئة التدريس. فى مثل هذه الحالة يمكن استخدام تعليمة الفرق التالية التى تربط بين نتائج تعليمتى اختيار:

```
SELECT FACULTY_ID
FROM FACULTY_T
WHERE SALARY > 35000
MINUS
SELECT FACULTY_ID
FROM QUALIFICATION_T
GROUP BY FACULTY_ID
HAVING COUNT(*) >= 2
ORDER BY FACULTY_ID;
```

وتكون نتيجة تعليمة الاختيار الأولى التى تظهر أرقام هيئة التدريس الذين يتقاضون أكثر من ٣٥,٠٠٠ ألفاً كما يلى:

```
FACULTY_
--------
320
330
420
500
540
640
660
770
800
810
```

أمـــا نتيجة تعليمة الاختيـــار الثانية فتظهر أرقام أعضاء هيئـــة التدريس المؤهلين لتدريس مادتين دراسيتين فأكثر، كما يلي:

```
FACULTY_
--------
200
220
320
810
```

ونظـــراً لكون نتائج كلتـــا عمليتى الاختيار الســـابقتين متوافقتين من حيث الاتحاد (تحتويان على العدد نفســـه من الحقول ومن النوعية نفســـها من البيانات) فإنه يمكن اســـتخدام عملية الفرق بينهما، ومن ثم ترتيـــب النتيجة تصاعدياً وفقاً لأرقام أعضاء هيئة التدريس، كما توضح النتيجة النهائية للتعليمة، وهى كما يلي:

```
FACULTY_
--------
330
420
500
540
640
660
770
800
```

وتتكـــون نتيجة تعليمة الفرق الســـابقة من ثمانية أرقام تمثـــل أرقام أعضاء هيئة التدريس الذين يتقاضون أكثر من ٣٥,٠٠٠ ألفاً ومؤهلين لتدريس مواد دراســـية يقل عددها عن مادتين دراسيتين.

الفصل الثامن

لغة الاستفسار البنائية - الجزء الثانى

يستكمل هذا الفصل شرح مكونات لغة الاستفسار البنائية بحيث خصص الجزء الأول منه لاستكمال شــرح تعليمة الاختيار، عندما تقوم التعليمة بالتعامل مع أكثر من جدول فى آن واحد، ولشــرح بقيــة تعليمات لغة معالجة البيانات وهــى: الإضافة، والحذف، والتحديث، أما الجزء الثانى فقد خصص لشرح تعليمات لغة التحكم فى البيانات.

٨-١ الضرب الكرتيزى وربط الجداول فى تعليمة الاختيار (Cartesian Product and Joining Tables):

إن عدم وجود عبارة شرط الاسترجاع (WHERE) ضمن تعليمة الاختيار (SELECT) يعنى عدم وجود أية شروط على نتيجة التعليمة، ويعنى هذا أن جميع سجلات الجدول المــدرج فى عبارة مصــدر الاســترجاع (FROM) مؤهلة لتكون ضمــن نتيجة التعليمة (كما أوضحنا فى الفصل الســابق عند شــرح تعليمة الاختيــار)، وعندما يتم تحديد أكثر من جدول ضمن عبارة مصدر الاســترجاع دون وجود عبارة شــرط الاسترجاع، فــإن نتيجــة التعليمة تكون الضرب الكرتيزى لســجلات الجــداول المدونة فى عبارة مصدر الاســترجاع، ويعنى هذا أن جميع التوليفات بين ســجلات الجداول ســتكون ضمن نتيجة تعليمة الاختيار، فعلى ســبيل المثــال، لنفترض وجود جدول أعضاء هيئة التدريس (Faculty_T) وجدول المجموعات الدراسية (SECTION_T) ضمن عبارة مصدر الاسترجاع، كما يلى:

```
SELECT *
FROM FACULTY_T, SECTION_T;
```

ســيكون عدد السجلات المســترجعة عند تنفيذ التعليمة السابقة أربعمائة سجل؛ وذلك لكون كل ســجل من سجلات أعضاء هيئة التدريس (وعددها عشرون) سيرتبط بكل ســجل من سجلات المجموعات الدراســية (وعددها عشرون أيضاً)، وعلى الرغم من أن التعليمة الســابقة بشكلها السابق قلما تستخدم على أرض الواقع لكون نتيجتها ليست ذات معنى، إلا أنها تعد أساساً لعمليات الربط بين الجداول فى لغة الاستفسار

البنائية، وعمليات الربط هى الوسيلة الوحيدة التى تمكننا من الانتقال من جدول إلى آخر وربط البيانات الموجودة فى سجلات الجداول المختلفة، وتتم عملية الربط من خلال استخدام عوامل المقارنة بين حقول الجداول قيد الربط، فعلى سبيل المثال، لمعرفة أسماء الطلبة وأرقام المواد الدراسية التى سجلوا فيها ونتائجهم فى هذه المواد، يتم ربط جدول التسجيل (ENROLLMENT_T) مع جدول الطلبة (STUDENT_T)، كما يلى:

```
SELECT FName, LName, Course_ID, Grade
FROM STUDENT_T, ENROLLMENT_T
WHERE STUDENT_T.STUDENT_ID = ENROLLMENT_T.STUDENT_ID;
```

ويمكن فهم طريقة عمل التعليمة السابقة كما يلى:

١- إجراء عملية الضرب الكرتيزى بين جدول المواد المسجلة (ENROLLMENT_T)، وجدول الطلبة (STUDENT_T)، ويكون ناتج هذه العملية جميع توليفات السجلات الموجودة فى جدول المواد المسجلة وجدول الطلبة، وتكون حقول السجلات الناتجة من هذه العملية عبارة عن جميع حقول الجدول الأول متبوعة بجميع حقول الجدول الثانى.

٢- يطبق عامل المقارنة «يساوى» (=) على ناتج عملية الضرب الكرتيزى بحيث تكون قيمة الحقل «رقم الطالب» الذى تم جلبه من جدول الطلبة (وهو يمثل المفتاح الرئيسى لجدول الطلبة) مساوياً لحقل «رقم الطالب» الذى تم جلبه من جدول المواد المسجلة (وهو يمثل مفتاحاً خارجياً فى جدول المواد المسجلة، وفى الوقت نفسه يُعدُّجزءاً من المفتاح الرئيسى للجدول)، وعند تساوى هذين الحقلين لسجل ما فى ناتج عملية الضرب الكرتيزى، يكون هذا السجل أحد السجلات الناتجة من عملية الاختيار.

٣- بعد معرفة السجلات الناتجة من عملية الربط، أى العمليتن السابقتين، يتم اختيار الحقول المطلوبة (وعددها أربعة حقول) حسب ورودها فى تعليمة الاختيار.

وبناءً على ذلك، يكون ناتج التعليمة السابقة كما يلى:

```
FNAME        LNAME        COURSE_     GRADE
------------ ------------ ------- ----------
Saleh        Alhamad      CHEM101          4
Abdullah     Aloufi       CHEM101          3
Khalid       Alsultan     CHEM101          4
Salem        Algamdi      CHEM101          3
Mishal       Alyousef     CHEM101          1
Saleh        Alhamad      CS101            2
Mishal       Alyousef     CS101            4
Saleh        Alhamad      CS102            3
Mishal       Alyousef     CS102            4
Saleh        Alhamad      ENGL101          3
Abdullah     Aloufi       ENGL101          4
Salem        Algamdi      ENGL101          4
Mishal       Alyousef     ENGL101          4
Saleh        Alhamad      ENGL102          1
Mishal       Alyousef     ENGL102          4
Saleh        Alhamad      MATH101          3
Abdullah     Aloufi       MATH101          2
Salem        Algamdi      MATH101          0
Mishal       Alyousef     MATH101          2
Saleh        Alhamad      MATH102          2
Mishal       Alyousef     MATH102          0
Saleh        Alhamad      STAT101          2
Mishal       Alyousef     STAT101          3
```

كما يمكن وضع أية شروط على ناتج عملية الربط، أو ترتيب نتائجها وفقاً لترتيب معين، فعلى سبيل المثال، لمعرفة أسماء الطلبة الذين درسوا فى مواد الحاسب الآلى أو مواد الرياضيات، ودرجاتهم، وترتيب النتيجة تصاعدياً حسب الاسم الأول للطلبة، يمكن استخدام التعليمة التالية:

```
SELECT FName, LName, Course_ID, Grade
FROM STUDENT_T, ENROLLMENT_T
WHERE STUDENT_T.STUDENT_ID = ENROLLMENT_T.STUDENT_ID
AND (COURSE_ID LIKE 'CS%' OR COURSE_ID LIKE 'MATH%')
ORDER BY FName ASC;
```

وتكون نتيجة التعليمة السابقة، كما يلى:

```
FNAME         LNAME         COURSE_     GRADE
------------  ------------  -------  ----------
Abdullah      Aloufi        MATH101           2
Mishal        Alyousef      MATH101           2
Mishal        Alyousef      CS101             4
Mishal        Alyousef      CS102             4
Mishal        Alyousef      MATH102           0
Saleh         Alhamad       MATH101           3
Saleh         Alhamad       CS102             3
Saleh         Alhamad       CS101             2
Saleh         Alhamad       MATH102           2
Salem         Algamdi       MATH101           0
```

وفى هذه الحالة يمكن فهم طريقة عمل التعليمة السـابقة، بالإضافة إلى ما سـبق أعلاه فى (١) و(٢)، كما يلى:

٣- بعد تحديد السـجلات المؤهلة من عملية الربط، تطبق عليها الشروط الواردة فى عبارة شرط الاختيار.

٤- يتـم اختيـار الحقول المطلوبـة (وعددها أربعة حقول) حسـب ورودها فى تعليمة الاختيار.

٥- يتم ترتيب سجلات النتيجة حسب الحقول الواردة فى عبارة «ترتيب حسب» (Order By) وحسب ترتيب الحقول فى العبارة (فى حالة الترتيب وفق أكثر من حقل).

وتسمى عملية الربط التى يستخدم فيها عامل المساواة (=) بعملية «ربط التساوي» (Equi-Join)، إلا أنه يمكن استخدام العوامل الأخرى وهى: {<, >, <>, =<, =>}، وتمثل العمليـة فى هذه الحالة ما يعـرف بالربط العـام (Theta-Join)، ونظراً لأهمية عملية الربـط بين الجـداول المختلفة للحصول على بيانات مترابطـة منطقياً فيما بينها فى النموذج العلاقى، توفر لغة الاستفسار البنائية عدداً من عبارات الربط، بالإضافة إلى اسـتخدام الربط بالطريقة السـابقة، ومن هذه العبارات عبارة «الربط» (JOIN) التى يمكن استخدامها لتمثيل عملية الربط السابقة كما يلى:

```
SELECT FName, LName, Course_ID, Grade
FROM STUDENT_T JOIN ENROLLMENT_T
    ON STUDENT_T.STUDENT_ID = ENROLLMENT_T.STUDENT_ID
WHERE COURSE_ID LIKE 'CS%' OR COURSE_ID LIKE 'MATH%'
ORDER BY FName ASC;
```

ففى المثال السابق تم استخدام عبارة الربط ضمن عبارة مصدر الاختيار وتم تحديد الحقل الذى ستطبق من خلاله عملية الربط بعد كلمة (ON)، وتعد هذه الطريقة أسهل للفهم من الطريقة السابقة؛ لأنها لا تدمج بين شروط الاختيار وعمليات الربط ضمن عبارة شـــرط الاختيار، ولكنها تفصل بينهما من خـــلال إدراج عملية الربط فى عبارة مصـــدر الاختيار وإدراج شـــروط الاختيار ضمن عبارة شـــرط الاختيار، وتكون نتيجة العملية الســـابقة جدولاً واحداً يحتوى على جميع حقول جدول الطلبة متبوعة بجميع حقول جدول المواد الدراســـية المسجلة، كما يلاحظ ذكر اسم الجدول قبل اسم الحقل الذى ســـتطبق من خلاله عملية الربط، وذلك لإزالة الالتباس بين مســـميات الحقول خاصة أن اســـم هذا الحقـــل متكرر فى الجدولين، ويمكن اختصار أســـماء الجدولين لتقليص طول عملية الاختيار من خلال عملية إعادة تسميتهما كما يلى:

```
SELECT FName, LName, Course_ID, Grade
FROM STUDENT_T S JOIN ENROLLMENT_T E
                    ON S.STUDENT_ID = E.STUDENT_ID
WHERE COURSE_ID LIKE 'CS%' OR COURSE_ID LIKE 'MATH%'
ORDER BY FName ASC;
```

وتوفـــر لغة الاستفســـار البنائيـــة أنواعاً مختلفـــة من الربط مـــن ضمنها «الربط الطبيعـــى» (Natural Join) وأنواعاً من «الربط الخارجـــى» (Outer Join)، ولإيضاح فكرة الربط الطبيعى، لنفترض أننا نرغب فى إظهار جميع الحقول الناتجة بعد عملية ربط جدول المواد الدراســـية مع جدول الأقسام الدراسية، فى هذه الحالة، تستخدم تعليمة الاختيار التالية:

```
SELECT *
FROM COURSE_T C JOIN DEPARTMENT_T D
                 ON C.DEPARTMENT_ID = D.DEPARTMENT_ID;
```

وتكون نتيجة التعليمة السابقة ربط بيانات كل مادة دراسية ببيانات القسم الدراسى الذى يقدمها، كما يلى:

```
COURSE_ TITLE                                          UNITS DEPART DEPART NAME
------- ------------------------------------ ---------- ------ ------ ----------------------
CHEM101 CHEMISTRY (I)                                      3 CHEM   CHEM   Chemistry
CHEM102 CHEMISTRY (II)                                     3 CHEM   CHEM   Chemistry
CS101   JAVA PROGRAMMING                                   3 CS     CS     Computer Science
CS102   SOFTWARE ENGINEERING                               3 CS     CS     Computer Science
CS103   C/C++ PROGRAMMING                                  3 CS     CS     Computer Science
CS104   COMPUTER ARCHITECTURE                              3 CS     CS     Computer Science
CS105   INTRODUCTION TO DATABASE SYSTEMS                   3 CS     CS     Computer Science
EE101   ELECTRIC CIRCUITS                                  3 EE     EE     Electrical Engineering
EE102   ELECTRONICS (I)                                    3 EE     EE     Electrical Engineering
EE103   ELECTRONICS (II)                                   3 EE     EE     Electrical Engineering
EE104   COMMUNICATION NETWORKS                             4 EE     EE     Electrical Engineering
ENGL101 ENGLISH GRAMMAR                                    2 ENGL   ENGL   English Language
ENGL102 ENGLISH WRITING                                    3 ENGL   ENGL   English Language
ENGL103 TECHNICAL WRITING                                  3 ENGL   ENGL   English Language
MATH101 INTRODUCTION To MATHEMATICS                        3 MATH   MATH   Mathematics
MATH102 DIFFERENTIAL EQUATIONS                             3 MATH   MATH   Mathematics
MATH103 CALCULUS (I)                                       3 MATH   MATH   Mathematics
MATH104 CALCULUS (II)                                      3 MATH   MATH   Mathematics
MATH106 ALGEBRA                                            4 MATH   MATH   Mathematics
MATH107 COMPUTER MATHEMATICS                               3 MATH   MATH   Mathematics
PHYS101 PHYSICS (I)                                        3 PHYS   PHYS   Physics
PHYS102 PHYSICS (II)                                       3 PHYS   PHYS   Physics
STAT101 INTRODUCTION TO STATISTICS                         3 STAT   STAT   Statistics
STAT102 ADVANCED STATISTICS                                3 STAT   STAT   Statistics
```

ويلاحظ فى النتيجة السابقة أن حقل رمز القسم الدراسى (الذى يمثل المفتاح الرئيسى لجدول الأقسام الدراسية، والمفتاح الخارجى لجدول المواد الدراسية) قد تكرر مرتين، ونظراً لأن تكرار بيانات هذا الحقل لا تضيف أية معلومة جديدة للنتيجة، فإنه يمكن إلغاء أحدهما دون الإخلال بالنتيجة، ويمكن إلغاء المتكرر من حقول الربط باستخدام «الربط الطبيعى» الذى يقوم بالربط بين جدولين، ضمنياً، حسب الأسماء المتطابقة للحقول فيهما (والتى قد تكون أكثر من زوج)، فعلى سبيل المثال، يمكن تنفيذ الاستفسار السابق باستخدام الربط الطبيعى كما يلى:

```
SELECT *
FROM COURSE_T NATURAL JOIN DEPARTMENT_T;
```

وتكون نتيجة العملية السابقة مكونة من خمسة حقول، عوضاً عن ستة حقول، بحيث تم إلغاء المتكرر وهو حقل رمز القسم الدراسى وإظهاره مرة واحدة فقط ضمن نتيجة العملية، وكأول حقل فى جدول النتيجة، كما يلى:

```
DEPART COURSE_ TITLE                                        UNITS NAME
------ ------- ---------------------------------- --------- ----------------------
CHEM   CHEM101 CHEMISTRY (I)                                  3 Chemistry
CHEM   CHEM102 CHEMISTRY (II)                                 3 Chemistry
CS     CS101   JAVA PROGRAMMING                               3 Computer Science
CS     CS102   SOFTWARE ENGINEERING                           3 Computer Science
CS     CS103   C/C++ PROGRAMMING                              3 Computer Science
CS     CS104   COMPUTER ARCHITECTURE                          3 Computer Science
CS     CS105   INTRODUCTION TO DATABASE SYSTEMS               3 Computer Science
EE     EE101   ELECTRIC CIRCUITS                              3 Electrical Engineering
EE     EE102   ELECTRONICS (I)                                3 Electrical Engineering
EE     EE103   ELECTRONICS (II)                               3 Electrical Engineering
EE     EE104   COMMUNICATION NETWORKS                         4 Electrical Engineering
ENGL   ENGL101 ENGLISH GRAMMAR                                2 English Language
ENGL   ENGL102 ENGLISH WRITING                                3 English Language
ENGL   ENGL103 TECHNICAL WRITING                              3 English Language
MATH   MATH101 INTRODUCTION To MATHEMATICS                    3 Mathematics
MATH   MATH102 DIFFERENTIAL EQUATIONS                         3 Mathematics
MATH   MATH103 CALCULUS (I)                                   3 Mathematics
MATH   MATH104 CALCULUS (II)                                  3 Mathematics
MATH   MATH106 ALGEBRA                                        4 Mathematics
MATH   MATH107 COMPUTER MATHEMATICS                           3 Mathematics
PHYS   PHYS101 PHYSICS (I)                                    3 Physics
PHYS   PHYS102 PHYSICS (II)                                   3 Physics
STAT   STAT101 INTRODUCTION TO STATISTICS                     3 Statistics
STAT   STAT102 ADVANCED STATISTICS                            3 Statistics
```

إن عمليـــة الربط الافتراضيـــة هى «الربط الداخلـــى» (Inner Join)، بمعنى أنه يتم إضافة سجل لنتيجة عملية الربط إذا توافر سجل فى أحد الجدولين وسجل مكافئ له فى الجدول الآخر وفق حقل (أو حقول) الربط، وبالنظر فى نتيجة المثال الذى يتطرق إلى معرفة أســماء الطلبة وأرقام المواد الدراســية التى سجلوا فيها ونتائجهم فى هذه المواد، أعلاه، فإن النتيجــة تحتوى على بعض الطلبة (وليس جميعهم)، وهؤلاء الطلبة هم الذين يوجد لهم ســجلات فى جدول المواد المســجلة، أما إذا أردنا معرفة أســماء الطلبة وأرقام المواد الدراســية التى ســجلوا فيها ونتائجهم فــى هذه المواد بالإضافة إلى الطلبة الذين لم يقوموا بتســجيل مواد دراســية، فإننا نستخدم «الربط الخارجى الأيســر» (Left Outer Join)، وحتى نقلص حجم نتيجة العملية، سنشــترط كون الطالب تابعاً لتخصص الحاسب الآلى، فى هذه الحالة تكون تعليمة الاختيار كما يلى:

```
SELECT FName, LName, Course_ID, Grade
FROM STUDENT_T S LEFT OUTER JOIN ENROLLMENT_T E
                 ON S.STUDENT_ID = E.STUDENT_ID
WHERE S.Major = 'CS';
```

ويلاحــظ فى النتيجة التاليــة للتعليمة ظهور جميع طلبة الحاســب الآلى، وظهور نتيجة المواد الدراســية للمســجلين منهم فى مواد دراسية، أما بالنسبة لغير المسجلين (وعددهم واحد فقط) فقد تم إدراج قيم غير معرفة ضمن حقول المواد المسجلين فيها،

ويعنى هذا أن نتيجة عملية «الربط الخارجى الأيسر» ستخرج ضمن نتائجها جميع الحقول المحددة فى تعليمة الاختيار **لجميع سجلات الجدول الأيسر** للعملية (وهو جدول الطلبة) بغض النظر عن وجود ما يطابقها فى الجدول الأيمن.

```
FNAME        LNAME        COURSE_  GRADE
------------ ------------ ------- ----------
Saleh        Alhamad      CHEM101        4
Mishal       Alyousef     CHEM101        1
Saleh        Alhamad      CS101          2
Mishal       Alyousef     CS101          4
Saleh        Alhamad      CS102          3
Mishal       Alyousef     CS102          4
Saleh        Alhamad      ENGL101        3
Mishal       Alyousef     ENGL101        4
Saleh        Alhamad      ENGL102        1
Mishal       Alyousef     ENGL102        4
Saleh        Alhamad      MATH101        3
Mishal       Alyousef     MATH101        2
Saleh        Alhamad      MATH102        2
Mishal       Alyousef     MATH102        0
Saleh        Alhamad      STAT101        2
Mishal       Alyousef     STAT101        3
Ghanim       Alhmoud
```

أما عملية «الربط الخارجى الأيمن» (Right Outer Join) فتعمل عكس عملية «الربط الخارجى الأيسر»، ويعنى هذا إظهار جميع الحقول المحددة فى تعليمة الاختيار **لجميع سجلات الجدول الأيمن** بغض النظر عن وجود ما يطابقها فى الجدول الأيسر من سجلات. فعلى سبيل المثال، لمعرفة المواد التابعة لقسم الحاسب الآلى ("CS") التى تم تسجيل بعض الطلبة فيها، وكذلك المواد الدراسية التابعة لقسم الحاسب الآلى التى لم يسجل فيها أى طالب مع إظهار أرقام الطلبة المسجلين فى المواد التى تم التسجيل فيها، يمكن استخدام «الربط الخارجى الأيمن» كما يلى:

```
SELECT Student_ID, C.Course_ID, C.Title
FROM ENROLLMENT_T E RIGHT OUTER JOIN COURSE_T C
                ON E.COURSE_ID = C.COURSE_ID
WHERE C.COURSE_ID LIKE 'CS%';
```

ويلاحظ فى النتيجة التالية للتعليمة ظهور مواد الحاسب الآلى كافة، وظهور أرقام الطلبة للمواد التى تم التسجيل فيها، أما بالنسبة للمواد التى لم يسجل فيها أى طالب، فتظهر المادة الدراسية وقد تم إدراج قيم غير معرفة فى حقل رقم الطالب.

```
STUDENT_ COURSE_ TITLE
-------- ------- ------------------------------------
19992020 CS101   JAVA PROGRAMMING
20001111 CS101   JAVA PROGRAMMING
19992020 CS102   SOFTWARE ENGINEERING
20001111 CS102   SOFTWARE ENGINEERING
         CS103   C/C++ PROGRAMMING
         CS104   COMPUTER ARCHITECTURE
         CS105   INTRODUCTION TO DATABASE SYSTEMS
```

بالإضافة إلى الربط الخارجى الأيمن والربط الخارجى الأيسر، توفر لغة الاستفسار البنائية «الربط الخارجى الكامل» (Full Outer Join)، وعند استخدام الربط الخارجى الكامل تكون النتيجة إظهار حقول السجلات التى تتطابق فى حقول الربط، وإظهار حقول السجلات فى الجدول الأيسر التى لا يوجد ما يطابقها فى الجدول الأيمن، وإظهار حقول سجلات الجدول الأيمن التى لا يوجد ما يطابقها فى الجدول الأيسر، ويتم استكمال بقية الحقول فى الجدول الناتج بتعبئتها بالقيمة غير المعرفة (NULL) لحالات عدم التطابق، سواء من قبل سجلات الجدول الأيمن أو الجدول الأيسر.

كما يمكن استخدام كلمة «طبيعى» (Natural) للربط بين جدولين، ضمنياً، من خلال الحقول المتطابقة فى الجدولين بالإضافة إلى إزالة حقل (أو حقول) الربط المتكررة من نتيجة تعليمة الاختيار، فعلى سبيل المثال يستخدم «الربط الطبيعى الأيسر» (Natural Left Outer Join) عند الرغبة فى إجراء الربط الأيسر بين جدولين وإظهار الحقول المتكررة مرة واحدة فقط ضمن النتيجة النهائية للتعليمة. توفر لغة الاستفسار البنائية أيضاً عبارة «الضرب الكرتيزى» (Cross Join) لاستخدامها بشكل ظاهر عوضاً عن استخدام مسميات الجداول فقط فى عبارة مصدر الاسترجاع، وعلى الرغم من أن جميع تعليمات الربط السابقة يمكن صياغتها من خلال العبارات الأخرى التى توفرها لغة الاستفسار البنائية، إلا أن وجود هذه العبارات ضمن لغة الاستفسار البنائية يسهل عملية قراءة المقصود منها مقارنة باستخدام تعليمات قد تكون أكثر تعقيداً فى فهمها للحصول على النتيجة نفسها، بالإضافة إلى ذلك، فإن وجود هذه العبارات يسهل، وبشكل كبير، كتابة تعليمات الربط المناسبة فى لغة الاستفسار البنائية.

من الممكن أيضاً استخدام تعليمات الربط بشكل متداخل، أو ربط أكثر من جدول فى الوقت نفسه، فعلى سبيل المثال، لمعرفة أعضاء هيئة التدريس الذين يعملون فى قسم الحاسب الآلى بالإضافة لمسميات المواد الدراسية المؤهلين لتدريسها، يجب ربط

جدول أعضاء هيئة التدريس (FACULTY_T) بجدول المواد الدراسـية من خلال جدول المؤهلات التدريسية (QULIFICATION_T)، كما يلى:

```
SELECT FName, LName, Title Qualified_to_Teach
FROM FACULTY_T NATURAL JOIN QUALIFICATION_T
                NATURAL JOIN COURSE_T
WHERE DEPARTMENT_ID = 'CS';
```

وتكون نتيجة التعليمة السابقة كما يلى:

```
FNAME        LNAME        QUALIFIED_TO_TEACH
------------ ------------ --------------------------------
Saleh        Aleesa       JAVA PROGRAMMING
Mohammed     Alhamad      SOFTWARE ENGINEERING
Mohammed     Alhamad      C/C++ PROGRAMMING
Ghanim       Alghanim     COMPUTER ARCHITECTURE
Ibraheem     Alsaleh      INTRODUCTION TO DATABASE SYSTEMS
```

أما إذا أردنا معرفة المجموعات الدراسـية التابعة لقسـم الحاسـب الآلى وأسماء الطلبة المسجلين فيها، متضمناً ذلك المجموعات التى لم يسجل فيها أى طالب، فيمكن استخدام التعليمة التالية:

```
SELECT FName, LName, Course_ID, Section_No, Year, Semester
FROM (((STUDENT_T NATURAL JOIN ENROLLMENT_T)
                NATURAL RIGHT OUTER JOIN SECTION_T)
                NATURAL JOIN COURSE_T)
WHERE DEPARTMENT_ID = 'CS';
```

ولقد تم استخدام عملية الربط الطبيعى الأولى فى التعليمة السابقة لمعرفة الطلبة والمجموعات الدراسية التى سجلوا فيها، أما عملية الربط الثانية فهى ربط طبيعى أيمن يمكننا من معرفة المجموعات الدراسـية كافة ومعرفة الطلبة المسـجلين فيها متضمناً ذلك المجموعات التى لم يسـجل فيها أى طالب، أما عملية الربط الثالثة فتمكننا من ربط المجموعات الدراسية والطلبة المسجلين فيها بالمواد التى تتبعها هذه المجموعات، وعملية الربط هذه ضرورية لمعرفة القسـم الذى تتبعه المجموعات الدراسـية التى لا تتوافر إلا من خلال جدول المواد الدراسـية الذى سـيطبق عليه شرط الاختيار، وهو أن تتبع المجموعة لقسـم الحاسـب الآلى، كما يلاحظ فى التعليمة السابقة استخدام الأقـواس، وذلك لتحديد أولويات عمليات الربط حيث يتم تنفيذ العمليات الواقعة بين

الأقواس من الداخل للخارج، بمعنى أن العمليات المضمنة ضمن الأقواس الداخلية تتم قبـل الأقواس التى تحتويها وهكذا حتى يتم تنفيذ الأقــواس الخارجية، وتكون نتيجة التعليمة السابقة كما يلى:

```
FNAME        LNAME        COURSE_ SECTION_NO       YEAR SEMESTER
------------ ------------ ------- ---------- ---------- ----------
Saleh        Alhamad      CS101            1       2000 FALL
Mishal       Alyousef     CS101            2       2000 FALL
Saleh        Alhamad      CS102            1       2000 SPRING
Mishal       Alyousef     CS102            1       2000 SPRING
                          CS103            1       2000 SPRING
                          CS104            1       2001 FALL
                          CS105            1       2001 SPRING
```

كما يمكن الاســتغناء عن عملية الربط الثالثة فى التعليمة السابقة والحصول على النتيجــة نفســها إذا ما لاحظنا أن رمــز المادة الدراســية (Course_ID) (ضمن جدول المجموعات الدراسية) يدل على اسم القسم الدراسى الذى تتبعه المجموعة الدراسية، حيث إن رمز المادة الدراسية يتكون من رمز القسم متبوعاً برقم المادة الدراسية، وبناءً على ذلك يمكن صياغة التعليمة السابقة كما يلى:

```
SELECT FName, LName, Course_ID, Section_No, Year, Semester
FROM ((STUDENT_T NATURAL JOIN ENROLLMENT_T)
                    NATURAL RIGHT OUTER JOIN SECTION_T)
WHERE COURSE_ID LIKE 'CS%';
```

تمكن لغة الاستفســـار البنائية أيضاً من ربط الجدول مع نفســه من خلال إعطائه مسميات مختلفة داخل تعليمة الاختيار، وفى هذه الحالة، يمكن اعتبار (أو تصور) كل مسمى يعطى للجدول على أساس أنه نسخة مختلفة للجدول نفسه، فعلى سبيل المثال، لمعرفة المواد الدراســية المؤهل لتدريسها أكثر من عضو هيئة تدريس، يمكن استخدام التعليمة التالية:

```
SELECT Q1.COURSE_ID, Q2.FACULTY_ID
FROM (QUALIFICATION_T Q1 JOIN QUALIFICATION_T Q2
                    ON Q1.COURSE_ID = Q2.COURSE_ID )
WHERE Q1.FACULTY_ID <> Q2.FACULTY_ID
GROUP BY Q1.COURSE_ID, Q2.FACULTY_ID;
```

ففى التعليمة السابقة تمت إعادة تسمية جدول المؤهلات التدريسية لأعضاء هيئة التدريس (QUALIFICATION_T) مرتين (مرة بمســمى Q1 ومرة بمسمى Q2) للحصول

على نسختين (تصوريتين) من جدول المؤهلات التدريسيه، وبعد ذلك تم ربط الجدولين من خلال الحقل المشترك «رقم المادة الدراسية» (Course_ID)، وحتى يتم التخلص من السجلات الناتجة بعد عملية الربط التى يكون فيها رقم عضو هيئة التدريس مكرراً (فى حقلين من حقول الجدول الناتج)، تمت إضافة شرط عدم التساوى فى عبارة شرط الاسترجاع، ومعنى هذا أن السجلات التى يتكرر فيها رقم عضو هيئة التدريس نفسه هى مواد دراسية مؤهل لتدريسها عضو هيئة تدريس واحد، ويجب حذفها من نتيجة التعليمة، أما بقية السجلات فهى سجلات لا يتكرر فيها رقم عضو هيئة التدريس نفسه؛ مما يعنى أنها تَدرس من قبل أساتذة مختلفين، وحتى يتم إظهار عضو هيئة التدريس الذى يقوم بتدريس مادة يقوم بتدريسها أعضاء هيئة تدريس آخرون مرة واحدة فقط ضمن نتيجة العملية، تم استخدام عبارة «التجميع حسب» (Group By)، التى يجب فيها استخدام الحقول المختارة كافة ضمن تعليمة الاختيار وهى حقل رقم المادة الدراسية، وحقل رقم عضو هيئة التدريس، وتكون نتيجة التعليمة السابقة كما يلى:

```
COURSE_ FACULTY_
------- --------
PHYS101 710
PHYS101 770
STAT101 600
STAT101 660
```

أما إذا أردنا معرفة أسماء أعضاء هيئة التدريس وعدم الاكتفاء بأرقامهم، فيمكن الربط مع جدول أعضاء هيئة التدريس (بالإضافة إلى الربط السابق لجدول المؤهلات التدريسية مع نفسه)، كما يلى:

```
SELECT Q1.COURSE_ID, F.FName, F.LName, Q2.FACULTY_ID
FROM ((QUALIFICATION_T Q1 JOIN QUALIFICATION_T Q2
                    ON  Q1.COURSE_ID = Q2.COURSE_ID)
                    JOIN FACULTY_T F
                    ON Q2.FACULTY_ID = F.FACULTY_ID)
WHERE Q1.FACULTY_ID <> Q2.FACULTY_ID
GROUP BY Q1.COURSE_ID, Q2.FACULTY_ID, F.FName, F.LName;
```

وتكون نتيجة التعليمة السابقة كما يلى:

```
COURSE_ FNAME        LNAME        FACULTY_
------- ------------ ------------ --------
PHYS101 Mahmood      Alsalem      710
PHYS101 Sultan       Aljasir      770
STAT101 Turki        Alturki      600
STAT101 Saud         Alkhalifa    660
```

٨-٢ الاستفسارات المتداخلة (Nested Queries):

تتطلب بعض عمليات الاختيار الحصول على قيم من قاعدة البيانات، ومن ثم استخدام هذه القيم فى عوامل مقارنة ضمن شرط الاختيار، ويمكن تكوين مثل عمليات الاختيار هذه من خلال ما يعرف بالاستفسارات المتداخلة التى يحتوى شرط الاسترجاع فيها على عمليات استفسار أخرى، وتسمى عملية الاستفسار الموجودة فى عبارة الشرط (WHERE) بعملية الاستفسار الداخلية (Inner Query)، فى حين تسمى تعليمة الاستفسار التى تحتويها بعملية الاستفسار الخارجية (Outer Query)، فعلى سبيل المثال، لو أردنا معرفة أعضاء هيئة التدريس الذين تزيد رواتبهم على متوسط رواتب أعضاء هيئة التدريس فإنه يمكننا ذلك من خلال تعليمتى استفسار، الأولى للحصول على متوسط رواتب أعضاء هيئة التدريس وهى كالتالى:

```
SELECT AVG(SALARY)
FROM FACULTY_T;
```

وتكون نتيجة التعليمة السابقة كما يلى:

```
AVG(SALARY)
-----------
      36940
```

أما التعليمة الثانية فستخدم النتيجة السابقة لمعرفة أعضاء هيئة التدريس الذين تزيد رواتبهم عن المتوسط الذى سبق حسابه أعلاه، كما يلى:

```
SELECT *
FROM FACULTY_T
WHERE SALARY > 36940;
```

وتكون نتيجة التعليمة السابقة، كما يلى:

```
FACULTY_ FNAME        LNAME        PHONE_NO     SALARY DOB       DEPART
-------- ------------ ------------ -------- ---------- --------- ------
320      Mohammed     Alhamad      454-5412      44000 13-MAY-65 CS
330      Ghanim       Alghanim     456-2234      44500 12-AUG-69 CS
420      Saleh        Alghamdi     454-2233      44600 13-FEB-69 CHEM
540      Salem        Alhamad      456-3304      40000 11-SEP-72 ENGL
640      Fahad        Alzaid       456-3322      44300 12-MAY-71 STAT
660      Saud         Alkhalifa    454-9856      44900 13-AUG-72 STAT
770      Sultan       Aljasir      456-3212      43300 13-MAY-70 PHYS
800      Ali          Albader      456-7812      45300 22-JUN-66 EE
810      Saad         Alzhrani     454-5578      44200 17-OCT-67 EE
```

إلا أنه باستخدام تعليمات الاستفسار المتداخلة يمكن دمج التعليمتين السابقتين، للحصول على النتيجة نفسها، ضمن تعليمة استفسار واحدة كما يلى:

```
SELECT *
FROM FACULTY_T
WHERE SALARY > (SELECT AVG(SALARY)
                FROM FACULTY_T);
```

ويمكن تصور طريقة عمل الاستفسار المتداخل السابق كما يلى:

١- تنفذ تعليمة الاختيار الداخلية أولاً.

٢- استخدام نتيجة تعليمة الاختيار الداخلية فى تنفيذ عملية الاختيار الخارجية.

ويعد الاستفسار الفرعى السابق من الاستفسارات الفرعية التى تعيد قيمة واحدة فقط، والتى يمكن استخدام أى من عوامل المقارنة التالية معها: {<>، > ، < ، = ، =< ، =>}، وعندما يتطلب الأمر استخدام أكثر من عملية استفسار داخلية، يمكن ربط الاستفسارات الداخلية من خلال العوامل المنطقية «أو» (OR)، «و» (AND)، فعلى سبيل المثال، لمعرفة بيانات أعضاء هيئة التدريس الذين تنحصر رواتبهم بين المتوسط العام للرواتب بعد زيادته بنسبة عشرة فى المائة، والمعدل العام بعد إنقاصه بنسبة عشرة فى المائة، يمكن استخدام الاستفسار المتداخل التالى:

```
SELECT *
FROM FACULTY_T
WHERE SALARY <= (SELECT AVG(SALARY)
                 FROM FACULTY_T) * 1.1
   AND SALARY >= (SELECT AVG(SALARY)
                 FROM FACULTY_T) * 0.9;
```

وتكون نتيجة الاستفسار السابق كما يلى:

```
FACULTY_ FNAME        LNAME        PHONE_NO  SALARY DOB       DEPART
-------- ------------ ------------ -------- ---------- --------- ------
200      Khalid       Aloufi       454-2341   35000 22-MAY-63 MATH
400      Ahmad        Alotaibi     454-4563   33900 17-MAY-71 CHEM
500      Yahya        Khorshid     456-2221   36700 12-MAR-65 ENGL
540      Salem        Alhamad      456-3304   40000 11-SEP-72 ENGL
560      Salman       Albassam     454-7865   33800 13-SEP-68 ENGL
850      Ahmad        Alsabti      456-0120   33900 15-APR-73 EE
```

عند اســتخدام الاستفسارات المتداخلة يجب توخى الدقة فى حالة تكرار مسميات الحقول فــى الجدول المســتخدم فى الاستفســار الداخلى والجدول المســتخدم فى الاستفســار الخارجى؛ إذ إنه يفضل دائماً ذكر اســم الجدول الذى يتبعه الحقل لإزالة الالتباس، والحالة الافتراضية فى لغة الاستفســار البنائية هى أن أى حقل متكرر دون ذكر اســم الجدول الذى يتبعه الحقــل، هو حقل الجدول التابع لأكثر الاستفســارات تداخلاً (Innermost Query)، ويشــابه هذا الوضــع ما يعرف «بحدود المتغيرات» (Scope of Variables) فــى لغــات البرمجــة؛ إذ إن أى متغير فى دالــة (FUNCTION) أو إجراء (PROCEDURE) يقصــد به المتغير المعرف ضمن الدالة أو الإجراء، وليس المتغير الذى يتبع للجزء من البرنامج الذى قام بتنشيط الدالة أو الأجراء.

٨-٢-١ العوامل العلاقية IN, ANY, ALL:

يلاحظ فى الاستفســارات الداخلية الســابقة أنها تقوم بإعــادة قيمة واحدة فقط، ولأن الحالة العامة لنتائج الاستفســارات فى لغة الاستفســار البنائية هى إعادة جداول تحتوى على مجموعات من القيم، وليس قيمة واحدة فقط؛ فإن لغة الاستفسار البنائية توفر ثلاثة عوامل علاقية أخرى للتعامل مع مجموعات من القيم، بالإضافة إلى عوامل المقارنة أعلاه، التى تسبق الاستفسارات الفرعية، وهذه العوامل هى: (IN, ANY, ALL).

يســتخدم العامل العلاقى (ALL) لمقارنة الحقل (أو التعبير الحسابى) قيد التحقق فى عبارة الشــرط لتعليمة الاستفسار الخارجية مع كافة القيم الناتجة من الاستفسار الداخلى، ويجب أن يســبق هذا العامل أحد عوامل المقارنة، فعلى سبيل المثال، لمعرفة أعضاء هيئة التدريس الذين تزيد رواتبهم عن متوسطات المرتبات فى أقسام الجامعة كافة، وترتيب النتيجة تصاعدياً حســب أســماء عائلات أعضاء هيئة التدريس، يمكن استخدام التعليمة التالية:

```
SELECT *
FROM FACULTY_T
WHERE SALARY > ALL (SELECT AVG(SALARY)
                    FROM FACULTY_T
                    GROUP BY DEPARTMENT_ID)
ORDER BY LName;
```

ويمكن فهم عمل التعليمة السابقة كما يلى:

١- حساب متوسط رواتب أعضاء هيئة التدريس فى كل قسم على حدة (حسب تعليمة الاستفسار الداخلية).

٢- لكل عضو هيئة تدريس فى جدول أعضاء هيئة التدريس (فى الاستفسار الخارجى)، تتم مقارنة راتبه بجميع متوسطات رواتب أعضاء هيئة التدريس لجميع أقسام الجامعة.

٣- إذا كان راتب عضو هيئة التدريس أعلى من جميع متوسطات الرواتب، يتم اختيار سجل عضو هيئة التدريس ضمن نتيجة التعليمة.

٤- ترتب السجلات التى تم اختيارها أبجدياً حسب أسماء عائلات أعضاء هيئة التدريس.

٥- يتم اختيار الحقول المحددة فى تعليمة الاختيار الخارجية لتمثل النتيجة النهائية للتعليمة (وهى جميع حقول جدول أعضاء هيئة التدريس فى هذه الحالة لاستخدام علامة «*»).

وتكون النتيجة النهائية للتعليمة كما يلى:

FACULTY_	FNAME	LNAME	PHONE_NO	SALARY	DOB	DEPART
800	Ali	Albader	456-7812	45300	22-JUN-66	EE
420	Saleh	Alghamdi	454-2233	44600	13-FEB-69	CHEM
330	Ghanim	Alghanim	456-2234	44500	12-AUG-69	CS
320	Mohammed	Alhamad	454-5412	44000	13-MAY-65	CS
770	Sultan	Aljasir	456-3212	43300	13-MAY-70	PHYS
660	Saud	Alkhalifa	454-9856	44900	13-AUG-72	STAT
640	Fahad	Alzaid	456-3322	44300	12-MAY-71	STAT
810	Saad	Alzhrani	454-5578	44200	17-OCT-67	EE

تجدر الإشارة إلى أن نتيجة عامل المقارنة (ALL) ستتحقق، بغض النظر عما يسبق هذا العامل من عوامل المقارنة مثل «=» أو «<>»، إذا كانت نتيجة الاستفسار الداخلى مجموعة خالية من القيم (Empty Set)، ويعنى هذا أن الحقل (أو التعبير الحسابى)

قيد التحقق منه بعامل المقارنة (ALL) سيحقق الشرط ويكون ضمن النتيجة النهائية للتعليمة إذا كانت نتيجة الاستفسار الداخلى مجموعة خالية من القيم.

على النقيض من عامل المقارنة (ALL)، فإن العامل العلاقى (ANY) يشترط أن يكون قيمة الحقل أو التعبير الحسابى قيد التحقق مقترناً بقيمة واحدة على الأقل من مجموعة القيم الناتجة من الاستفسار الداخلى، وكما هو الحال فى عامل المقارنة (ALL) يجب أن يسبق عامل المقارنة (ANY) أحد عوامل المقارنة التالية: {= , <> , > , < , >= , <=}، فعلى سبيل المثال، لمعرفة أعضاء هيئة التدريس الذين تزيد رواتبهم عن متوسطات المرتبات فى أى من أقسام الجامعة، وترتيب النتيجة تصاعدياً حسب أسماء عائلات أعضاء هيئة التدريس، يمكن استخدام التعليمة التالية:

```
SELECT *
FROM FACULTY_T
WHERE SALARY > ANY (SELECT AVG(SALARY)
                    FROM FACULTY_T
                    GROUP BY DEPARTMENT_ID)
ORDER BY LName;
```

وتكون نتيجة التعليمة السابقة كما يلى:

FACULTY_	FNAME	LNAME	PHONE_NO	SALARY	DOB	DEPART
800	Ali	Albader	456-7812	45300	22-JUN-66	EE
560	Salman	Albassam	454-7865	33800	13-SEP-68	ENGL
420	Saleh	Alghamdi	454-2233	44600	13-FEB-69	CHEM
330	Ghanim	Alghanim	456-2234	44500	12-AUG-69	CS
320	Mohammed	Alhamad	454-5412	44000	13-MAY-65	CS
540	Salem	Alhamad	456-3304	40000	11-SEP-72	ENGL
770	Sultan	Aljasir	456-3212	43300	13-MAY-70	PHYS
660	Saud	Alkhalifa	454-9856	44900	13-AUG-72	STAT
400	Ahmad	Alotaibi	454-4563	33900	17-MAY-71	CHEM
200	Khalid	Aloufi	454-2341	35000	22-MAY-63	MATH
850	Ahmad	Alsabti	456-0120	33900	15-APR-73	EE
710	Mahmood	Alsalem	456-3323	31900	19-FEB-73	PHYS
640	Fahad	Alzaid	456-3322	44300	12-MAY-71	STAT
810	Saad	Alzhrani	454-5578	44200	17-OCT-67	EE
500	Yahya	Khorshid	456-2221	36700	12-MAR-65	ENGL

إن نتيجة عامل المقارنة (ANY)، على النقيض من عامل المقارنة (ALL)، لا تتحقق، بغض النظر عما يسبق هذا العامل من عوامل المقارنة مثل «=» أو «<>»، إذا كانت نتيجة الاستفسار الداخلى مجموعة خالية من القيم (Empty Set)، ويعنى هذا أن الحقل (أو التعبير الحسابى) قيد التحقق منه بعامل المقارنة (ANY) لن يحقق الشرط، وبذلك

لن يكون ضمن النتيجة النهائية للتعليمة إذا كانت نتيجة الاستفسار الداخلى مجموعة خالية من القيم.

أما عامل المقارنة (IN) فيعمل على التحقق من أن قيمة الحقل أو التعبير الحسابى قيد التحقق هو من ضمن مجموعة القيم الناتجة من الاستفسار الداخلى، وبذلك فهو مكافئ لعامل المقارنة (ANY) مسبوقاً بعامل المقارنة (=) كما يلى: (ANY=)، ويعنى هذا أن عامل المقارنة (ANY) أشمل فى الاستخدام من عامل المقارنة (IN)، إلا أن عامل المقارنة (IN) لا يتطلب أن يسبق بعوامل مقارنة أخرى كما هو الحال فى حالة عامل المقارنة (ANY)، وعامل المقارنة (ALL)، فعلى سبيل المثال، لمعرفة أعضاء هيئة التدريس الذين لا يعملون فى قسم الهندسة الكهربائية ويتقاضون مرتبات تساوى أياً كان من أعضاء هيئة التدريس الذين يعملون فى قسم الهندسة الكهربائية، وترتيب النتيجة تصاعدياً حسب أسماء عائلات أعضاء هيئة التدريس، يمكن استخدام التعليمة التالية:

```
SELECT *
FROM FACULTY_T
WHERE SALARY IN (SELECT DISTINCT SALARY
                 FROM FACULTY_T
                 WHERE DEPARTMENT_ID = 'EE')
AND DEPARTMENT_ID <> 'EE'
ORDER BY LName;
```

وتكون نتيجة التعليمة السابقة كما يلى:

```
FACULTY_ FNAME        LNAME        PHONE_NO     SALARY DOB       DEPART
-------- ------------ ------------ -------- ---------- --------- ------
400      Ahmad        Alotaibi     454-4563      33900 17-MAY-71 CHEM
```

وتكون نتيجة التعليمة السابقة مكافئة لنتيجة استخدام عامل المقارنة (ANY) الواردة فى التعليمة التالية:

```
SELECT *
FROM FACULTY_T
WHERE SALARY = ANY (SELECT DISTINCT SALARY
                    FROM FACULTY_T
                    WHERE DEPARTMENT_ID = 'EE')
AND DEPARTMENT_ID <> 'EE'
ORDER BY LName;
```

كمـا يمكن أن يسـبق عامل المقارنة (IN) بالنفى ليصبـح (NOT IN)، ويعنى هذا أن الحقل أو التعبير الحسابى قيد التحقق سيكون من ضمن النتيجة النهائية للاستفسار الخارجى إذا لم يكن من ضمن مجموعة القيم الناتجة من الاستفسـار الداخلى قيمة تكافئ قيمة الحقل أو التعبير الحسابى، ويكافئ عامل المقارنة (NOT IN) عامل المقارنة (ALL) مسبوقاً بعامل المقارنة (<>) كما يلى: (<> ALL)، فعلى سبيل المثال، لمعرفة مواد الحاسب الآلى غير المنفذة فى فصل الربيع (SPRING) من عام ٢٠٠٠، يمكن استخدام التعليمة التالية:

```
SELECT *
FROM COURSE_T
WHERE COURSE_ID NOT IN (SELECT COURSE_ID
                        FROM SECTION_T
                        WHERE SEMESTER = 'SPRING' AND
                              YEAR = 2000)
AND COURSE_ID LIKE 'CS%';
```

تقوم تعليمة الاستفسـار الداخلية، أعلاه، باسـتخدام جدول المجموعات الدراسية لتحديـد المواد المنفذة كافة فـى فصل الربيع من عام ٢٠٠٠، أما تعليمة الاستفسـار الخارجيـة فتقوم بتحديد مواد الحاسـب الآلى غير المنفذة فـى فصل الربيع من عام ٢٠٠٠ باسـتخدام عامل المقارنة (NOT IN) مع مجموعة القيم الناتجة من الاستفسـار الداخلى، وتكون نتيجة التعليمة السابقة كما يلى:

```
COURSE_ TITLE                                UNITS DEPART
------- ------------------------------------ ---------- ------
CS101   JAVA PROGRAMMING                         3 CS
CS104   COMPUTER ARCHITECTURE                    3 CS
CS105   INTRODUCTION TO DATABASE SYSTEMS         3 CS
```

٨-٢-٢ الاستفسارات المتداخلة المتعددة المستويات:

تمكن لغة الاستفسار البنائية من تكوين استفسارات متداخلة بمستويات تزيد على مستويين، فعلى سـبيل المثال، لمعرفة أعضاء هيئة التدريس المؤهلين لتدريس أكثر من مـادة بحيث إن كل مادة من المواد المؤهلين لتدريسـها قد تم تنفيذها مرة واحدة على الأقل، يمكن استخدام الاستفسار ذى المستويات الثلاثة التالى:

```
SELECT *
FROM FACULTY_T
WHERE FACULTY_ID IN (SELECT FACULTY_ID
                     FROM QUALIFICATION_T
                     WHERE COURSE_ID IN (SELECT COURSE_ID
                                         FROM SECTION_T)
                     GROUP BY FACULTY_ID
                     HAVING COUNT(FACULTY_ID) > 1);
```

ويتم تنفيذ التعليمة السابقة كما يلى:

١- يتم تحديد أرقام المواد الدراسية المنفذة أولاً من خلال تنفيذ تعليمة الاستفسار الداخلية المعرف مصدر استرجاعها على أنه جدول المجموعات الدراسية.

٢- يتم تنفيذ تعليمة الاستفسار الأعلى فى المستوى والتى تحدد أرقام أعضاء هيئة التدريس المؤهلين لتدريس مواد تقع ضمن مجموعة المواد المنفذة، ويتم تجميع السجلات الناتجة حسب أرقام أعضاء هيئة التدريس، بعد ذلك يتم ترشيح المجموعات حسب عدد السجلات فى كل مجموعة بحيث أن كل مجموعة تمثل عدد تكرارات رقم عضو هيئة التدريس، وعندما يتكرر رقم عضو هيئة التدريس (فى المجموعة) يكون رقم عضو هيئة التدريس ضمن نتيجة الاستفسار الفرعي.

٣- يتم تنفيذ الاستفسار الأعلى فى المستوى (وهو الاستفسار الأخير) بحيث يتم إظهار جميع بيانات أعضاء هيئة التدريس الذين تقع أرقامهم ضمن مجموعة السجلات الناتجة من الاستفسار الفرعى الأدنى فى المستوى.

وتكون النتيجة النهائية للاستفسار السابق كما يلى:

```
FACULTY_ FNAME        LNAME        PHONE_NO     SALARY DOB       DEPART
-------- ------------ ------------ -------- ---------- --------- ------
200      Khalid       Aloufi       454-2341      35000 22-MAY-63 MATH
220      Fahad        Alhamid      456-7733      25900 07-OCT-70 MATH
320      Mohammed     Alhamad      454-5412      44000 13-MAY-65 CS
```

٨-٢-٣ الاستفسارات المتداخلة المرتبطة (Correlated Nested Queries):

تم شرح الاستفسارات المتداخلة التى تقوم لغة الاستفسار البنائية بتنفيذها مرة واحدة فقط، بمعنى أن كل استفسار داخلى يتم تنفيذه مرة واحدة وبمعزل عن الاستفسار الخارجى الذى يحتويه، ويعنى هذا أيضاً أن الاستفسارات الفرعية السابقة

تنفذ على أنها استفسارات مستقلة قائمة بذاتها دون أى ارتباط بسجلات جدول الاستفسار الخارجى الذى يحتويها، وبعد تنفيذ الاستفسار الداخلى تستخدم نتيجته للمقارنة مع كل سجل من سجلات الجدول المستخدم فى تعليمة الاستفسار الخارجية، إلا أنه فى الكثير من الأحيان تظهر حاجة لربط الاستفسار الداخلى بسجلات جدول (أو جداول) الاستفسار الخارجى، ويستدعى هذا عملية تنفيذ الاستفسار الداخلى لكل سجل من سجلات جدول الاستفسار الخارجى، وتتم عملية الارتباط هذه عندما يتم استخدام أحد حقول جداول الاستفسار الخارجى ضمن شرط الاسترجاع فى الاستفسار الداخلى، فى هذه الحالة يسمى الاستفساران (الداخلى والخارجى) مترابطين (Correlated)، وعلى النقيض من الاستفسارات غير المترابطة، يتم استخدام بيانات السجلات الموجودة فى جدول الاستفسار الخارجى فى تنفيذ الاستفسار الداخلى، وتتم عملية التنفيذ لكل سجل على حدة، فعلى سبيل المثال، لمعرفة أعضاء هيئة التدريس المؤهلين لتدريس مواد معينة ولم يقوموا حتى الآن بتدريسها على الرغم من تنفيذها، يمكن استخدام التعليمة التالية:

```
SELECT *
FROM FACULTY_T
WHERE FACULTY_ID IN (SELECT FACULTY_ID
                     FROM QUALIFICATION_T Q
                     WHERE COURSE_ID NOT IN (SELECT COURSE_ID
                                  FROM SECTION_T S
                                  WHERE Q.Faculty_ID = S. Faculty_ID)
                     AND COURSE_ID IN (SELECT COURSE_ID
                                  FROM SECTION_T));
```

ويمكن فهم طريقة تنفيذ التعليمة السابقة كما يلى:

١- تنفذ تعليمة الاستفسار الداخلية غير المرتبطة لمعرفة أرقام المواد الدراسية كافة التى تم تنفيذها فى الجامعة، وهذه التعليمة كما يلى:

```
SELECT COURSE_ID
FROM SECTION_T;
```

وتكون نتيجة التعليمة السابقة كما يلى:

```
COURSE_
-------
CHEM101
CHEM101
CS101
CS101
CS102
CS103
CS104
CS105
EE101
EE102
ENGL101
ENGL102
MATH101
MATH102
MATH103
MATH104
PHYS101
PHYS102
STAT101
STAT102
```

وعلى الرغم من تكرار بعض أرقام المواد الدراسية فى النتيجة، إلا أن هذا ليس ذا أهمية على النتيجة النهائية لهذا المثال.

٢- تنفــذ التعليمة المرتبطة الداخلية لمعرفة أرقام المواد الدراســية التى قام كل عضو هيئة تدريس بتدريســها، فعلى ســبيل المثال، لنفترض عضو هيئــة التدريس رقم «٦٦٠»، فى هذه الحالة لن ينتج أى ســجل بعد تنفيذ التعليمة المرتبطة؛ لأن عضو هيئة التدريس هذا لم يقم بتدريس أى مادة دراسية، ويكون شكل التعليمة المرتبطة فى هذه الحالة كما يلى:

```
SELECT COURSE_ID
FROM SECTION_T S
WHERE S. Faculty_ID; = '660'
```

٣- تنفــذ تعليمة الاستفســار الخارجية التى تدمج بين التعليمتين الســابقتين، لمعرفة أرقام أعضاء هيئة التدريس المؤهلين لتدريس مواد دراسية، ولكنهم لم يدرسوا هذه المواد المؤهلين لتدريسها على الرغم من تنفيذها، كما يلى:

```
SELECT FACULTY_ID
FROM QUALIFICATION_T Q
WHERE COURSE_ID NOT IN (SELECT COURSE_ID
                        FROM SECTION_T S
                        WHERE Q.Faculty_ID = S. Faculty_ID)
                        AND COURSE_ID IN (SELECT COURSE_ID
                                          FROM SECTION_T);
```

وتكون نتيجة التعليمة كما يلى:

```
FACULTY_
--------
770
660
```

ويلاحــظ ظهور رقم عضو هيئــة التدريس ٦٦٠، وذلك لكون هــذا العضو مؤهلاً لتدريس مادة الإحصاء (STAT101) حسب محتويات جدول المؤهلات التدريسية لأعضاء هيئة التدريس، ولكنه لم يقم بتدريســها حسب محتويات جدول المجموعات الدراسية حيــث قــد قام عضو هيئــة تدريس آخر (وهــو ذو الرقم ٦٠٠) بتدريــس هذه المادة، وكذلك هو الحال بالنسبة لعضو هيئة التدريس رقم ٧٧٠ المؤهل لتدريس مادة الفيزياء (PHYS101) ولكنه لم يقم بتدريســها على الرغم من تنفيذها حسب جدول المجموعات الدراسية حيث قام بتنفيذها عضو هيئة التدريس ذو الرقم ٧١٠.

٤- تنفذ تعليمة الاستفســار الخارجية الأخيرة لمعرفة بيانــات أعضاء هيئة التدريس ذات الأرقام الواردة ضمن النتيجة النهائية (الســابقة) للاستفسار الداخلى، وتكون نتيجة التعليمة كما يلى:

```
FACULTY_ FNAME        LNAME        PHONE_NO     SALARY DOB       DEPART
-------- ------------ ------------ -------- ---------- --------- ------
660      Saud         Alkhalifa    454-9856      44900 13-AUG-72 STAT
770      Sultan       Aljasir      456-3212      43300 13-MAY-70 PHYS
```

٨-٢-٣-١ العامل العلاقى EXISTS:

يستخدم عامل المقارنة (EXISTS) لاختبار نتيجة الاستفسار الفرعى من حيث خلوه من أى نتيجة (Empty Set)، فعندما يكون الاستفســار الفرعى خالياً من النتائج تصبح نتيجة العامل (EXISTS) خطأ (False)، وعندما تحتوى نتيجة الاستفســار الفرعى على

نتائج، تصبح نتيجـــة العامل (EXISTS) صح (True)، وفى الغالبية العظمى من الأحيان يســـتخدم العامل العلاقى (EXIXTS) مع الاستفسارات المتداخلة المرتبطة، فعلى سبيل المثال، لمعرفة أعضاء هيئة التدريس الذين قاموا بتدريس مواد دراسية، يمكن استخدام الاستفسار المتداخل المرتبط التالى:

```
SELECT *
FROM FACULTY_T F
WHERE EXISTS (SELECT *
              FROM SECTION_T S
              WHERE F.Faculty_ID = S. Faculty_ID);
```

يقوم الاستفســـار الســـابق بتحديد ســـجل أحد أعضاء هيئة التدريس من جدول أعضاء هيئة التدريس الوارد فى الاستفســـار الخارجي، ومن ثم يقوم باسترجاع جميع الســـجلات من جدول المجموعات الدراسية التى يوجد فيها رقم عضو هيئة التدريس الذى تم تحديده من ضمنها، بعد ذلك يتم التحقق من خلو نتيجة الاستفسار الداخلى من السجلات، وعندما تكون نتيجة الاستفسار الداخلى غير خالية من السجلات فإن هذا يعنى أن عضو هيئة التدريس قد قام بتدريس مواد دراسية، وخلاف ذلك يعنى أن عضو هيئة التدريس لم يقم بتدريس أية مادة دراســـية، وتكون نتيجة التعليمة السابقة كما يلى:

FACULTY_	FNAME	LNAME	PHONE_NO	SALARY	DOB	DEPART
200	Khalid	Aloufi	454-2341	35000	22-MAY-63	MATH
220	Fahad	Alhamid	456-7733	25900	07-OCT-70	MATH
310	Saleh	Aleesa	454-8932	30000	13-SEP-66	CS
320	Mohammed	Alhamad	454-5412	44000	13-MAY-65	CS
330	Ghanim	Alghanim	456-2234	44500	12-AUG-69	CS
340	Ibraheem	Alsaleh	454-1234	25000	20-JAN-70	CS
400	Ahmad	Alotaibi	454-4563	33900	17-MAY-71	CHEM
500	Yahya	Khorshid	456-2221	36700	12-MAR-65	ENGL
540	Salem	Alhamad	456-3304	40000	11-SEP-72	ENGL
600	Turki	Alturki	456-7891	27800	23-JUL-75	STAT
640	Fahad	Alzaid	456-3322	44300	12-MAY-71	STAT
710	Mahmood	Alsalem	456-3323	31900	19-FEB-73	PHYS
730	Mishal	Almazid	454-2343	29800	17-SEP-75	PHYS
800	Ali	Albader	456-7812	45300	22-JUN-66	EE
810	Saad	Alzhrani	454-5578	44200	17-OCT-67	EE

أمـــا إذا أردنـــا معرفة أعضاء هيئـــة التدريس الذين لم يقومـــوا بتدريس أية مادة دراسية، فيمكن استخدام الاستفسار المتداخل التالى الذى يستخدم فيه عامل المقارنة (NOT EXIXTS)، كما يلى:

```
SELECT *
FROM FACULTY_T F
WHERE NOT EXISTS (SELECT *
                  FROM SECTION_T S
                  WHERE F.Faculty_ID = S. Faculty_ID);
```

وتكون نتيجة الاستفسار السابق كما يلى:

```
FACULTY_ FNAME        LNAME        PHONE_NO     SALARY DOB       DEPART
-------- ------------ ------------ -------- ---------- --------- ------
420      Saleh        Alghamdi     454-2233      44600 13-FEB-69 CHEM
560      Salman       Albassam     454-7865      33800 13-SEP-68 ENGL
660      Saud         Alkhalifa    454-9856      44900 13-AUG-72 STAT
770      Sultan       Aljasir      456-3212      43300 13-MAY-70 PHYS
850      Ahmad        Alsabti      456-0120      33900 15-APR-73 EE
```

ومثــالاً آخــر، لنفترض أننا نرغب فى معرفة المواد الدراســية التى لــم تنفذ من قبل الجامعة الأهلية حتى الآن، للإجابة عن هذا الاستفسار يمكن استخدام التعليمة التالية:

```
SELECT *
FROM COURSE_T C
WHERE NOT EXISTS (SELECT *
                  FROM SECTION_T S
                  WHERE C.Course_ID = S. Course_ID);
```

وتكون نتيجة التعليمة السابقة كما يلى:

```
COURSE_ TITLE                                     UNITS DEPART
------- ---------------------------------------- ---------- ------
CHEM102 CHEMISTRY (II)                                    3 CHEM
EE103   ELECTRONICS (II)                                  3 EE
EE104   COMMUNICATION NETWORKS                            4 EE
ENGL103 TECHNICAL WRITING                                 3 ENGL
MATH106 ALGEBRA                                           4 MATH
MATH107 COMPUTER MATHEMATICS                              3 MATH
```

٨-٣ تعليمات الإضافة، والحذف، والتحديث (Insert, Delete and Update Statements):

توفر لغة الاستفســـار البنائية ثلاث تعليمـــات لتغيير محتوى قاعدة البيانات وهى: تعليمـــة الإضافة، وتعليمة الحذف، وتعليمة التحديث، وفيما يلى شـــرح كل واحدة من هذه التعليمات الثلاث.

٨-٣-١ تعليمة الإضافة:

إن أبسط صور تعليمة الإضافة هى إضافة سجل لجدول ما، فى هذه الحالة يجب ذكر اسم الجدول وقيم الحقول للسجل قيد الإضافة، كما يتوجب إدخال قيم الحقول وفق الترتيب نفسه الذى تم إدراجها فيه أثناء إنشاء الجدول، فعلى سبيل المثال، يمكن إضافة بيانات عضو هيئة تدريس جديد يعمل فى قسم الرياضيات كما يلى:

```
INSERT INTO FACULTY_T
VALUES ('205', 'Saleh', 'Altimimi', '454-2233', 35000, '22-MAY-1963', 'MATH');
```

وتجدر ملاحظة القيود فى أثناء عملية إدخال البيانات مثل المفتاح الرئيسى الذى يجب أن لا يتكرر مع سجل موجود أصلاً فى الجدول، وقيد القيم غير المعرفة (NOT NULL) حيث إن أى حقل مرتبط بهذا القيد يجب أن تدخل قيمة له وإلا فشلت عملية الإضافة، والقيد الفريد (UNIQUE) حيث إن أى حقل مرتبط بهذا القيد يجب أن تكون له قيمة غير متكررة مع سجل موجود أصلاً فى الجدول، على الرغم من أن قيمة الحقل من الممكن أن تكون غير معرفة (NULL) لأكثر من سجل واحد كما أسلفنا عن شرح قيود السجلات، وإلا فشلت عملية الإضافة أيضاً.

أما الشكل الثانى لتعليمة الإضافة فهو عند الرغبة فى إدخال قيم بعض الحقول وليس جميعها، ويعد هذا الشكل من التعليمة مفيداً جداً عندما يكون عدد حقول الجدول كبيراً جداً ونرغب فى إدخال بعض منها فقط، وفى هذه الحالة يجب ذكر أسماء الحقول التى سيتم إدخال قيم لها قبل إدراج القيم التى سيتم إدخالها، ونظراً لذكر أسماء الحقول ضمن عملية الإضافة، فإنه يمكن إدخالها بأى ترتيب شئنا مادامت القيم المدخلة متوافقة مع ترتيب الحقول، فعلى سبيل المثال، يمكن إدخال بيانات عضو هيئة تدريس جديد يعمل فى قسم الرياضيات كما يلى:

```
INSERT INTO FACULTY_T (FName, Faculty_ID, LName, Department_ID, DOB)
VALUES ('Mohamed', '207', 'Alsalem', 'MATH', '22-MAY-1963');
```

وعند استخدام الشكل السابق للتعليمة فإن أى حقل لم تدخل له قيمة ضمن تعليمة الإضافة ستكون قيمته غير معرفة أو ستأخذ القيمة الافتراضية فى حال ارتبط الحقل الذى لم تدخل قيمة له بقيمة افتراضية فى أثناء إنشاء الجدول، أما إذا أردنا إدخال قيمة غير معرفة لحقل ما، فتستخدم كلمة (NULL) لهذا الغرض، فمثلاً، يمكن إدخال القيمة غير المعرفة لحقل الراتب (SALARY)، كما يلى:

```
INSERT INTO FACULTY_T (FName, Faculty_ID, LName,  Department_ID, DOB,
Salary)
 VALUES ('Mohamed', '207',  'Alsalem', 'MATH', '22-MAY-1963', NULL);
```

ويوجد شكل ثالث لعملية الإضافة يستخدم لإضافة مجموعة من السجلات، تكون ناتجة من عملية اختيار، إلى جدول، فعلى سـبيل المثال لنفترض أننا نرغب فى إنشاء جـدول جديد بمسـمى (TEMP_T) يحتوى على ثلاثة حقـول: حقل يحتوى على رمز القسـم الدراسـى، وحقل يحتوى على عدد أعضاء هيئة التدريس فى القسـم، وحقل يحتـوى على مجموع رواتب أعضاء هيئة التدريس فى القسـم، فى هذه الحالة يمكن إنشاء الجدول كما يلى:

```
CREATE TABLE TEMP_T
(DEPARTMENT_ID      CHAR(6)         PRIMARY KEY,
FACULTY_NO          NUMBER,
TOTAL_SALARY        NUMBER);
```

وبعد إنشاء الجدول، يمكن إدخال البيانات إليه كما يلى:

```
INSERT INTO TEMP_T
SELECT DEPARTMENT_ID, COUNT(*) Faculty_No, SUM(SALARY) Total_Salary
FROM FACULTY_T
GROUP BY DEPARTMENT_ID;
```

وتكون محتويات الجدول الجديد (Temp_T) بعد تنفيذ العملية السابقة كما يلى:

```
DEPART FACULTY_NO TOTAL_SALARY
------ ---------- ------------
CHEM            2        78500
CS              4       143500
EE              3       123400
ENGL            3       110500
MATH            2        60900
PHYS            3       105000
STAT            3       117000
```

كما يمكن إدراج أسماء الحقول ضمن تعليمة الإدخال كما يلى:

```
INSERT INTO TEMP_T (DEPARTMENT_ID, FACULTY_NO, TOTAL_SALARY)
SELECT DEPARTMENT_ID, COUNT(*), SUM(SALARY)
FROM FACULTY_T
GROUP BY DEPARTMENT_ID;
```

وتجدر الإشـــارة إلى أنه ليس من الضـــرورى أن يكون الجدول المراد إدخال قيم له فارغاً من الســـجلات حتى يمكن اســـتخدام الشكل الســـابق للتعليمة، وإنما قد يكون الجدول محتوياً على ســـجلات قبل إدخال قيم جديدة فيه باســـتخدام الشكل السابق للتعليمة.

٨-٣-٢ تعليمة الحذف:

تستخدم تعليمة الحذف (DELETE) لحذف السجلات من الجداول، وتحتوى تعليمة الحذف على عبارة الشـــرط (WHERE) الشبيه بشـــرط الاسترجاع فى عبارة الاختيار (SELECT)، ويتـــم تنفيذ عملية الحذف على ســـجل واحد أو مجموعة من الســـجلات فى جدول واحد فقط، فعلى ســـبيل المثال، يمكن حذف سجل عضو هيئة التدريس ذى الرقم «٢٠٧» الذى تم إدخاله فى جدول أعضاء هيئة التدريس أعلاه، كما يلى:

```
DELETE FROM FACULTY_T WHERE FACULTY_ID = '207';
```

كما يمكن حذف مجموعة من الســـجلات عوضاً عن ســـجل محدد واحد حســـب الشـــرط الذى يتم تحديده فى عبارة الشـــرط، فمثلاً، يمكن حذف الأقسام الدراسية التـــى يقل عدد أعضاء هيئة التدريس فيها عن ثلاثـــة من الجدول المؤقت (TEMP_T) كما يلى:

```
DELETE FROM TEMP_T WHERE FACULTY_NO < 3;
```

أما إذا أردنا حذف الســـجلات كافـــة من جدول ما، فإنه يمكن اســـتخدام تعليمة الحذف دون استخدام عبارة الشرط، فمثلاً، يمكن حذف بقية السجلات الموجودة فى الجدول المؤقت (TEMP_T) كما يلى:

```
DELETE FROM TEMP_T;
```

وعلى الرغم من أننا قد قمنا بحذف الســـجلات كافـــة من الجدول المؤقت، إلا أن تعريف هيكل الجدول يســـتمر موجوداً ضمن مكونـــات قاعدة البيانات، ويعنى هذا أن تعليمة الحذف تقوم بحذف الســـجلات، سواء بشـــكل فردى أم مجموعات، ولكنها لا تقـــوم بإزالة هياكل الجدول، ولحذف هيكل الجدول تســـتخدم تعليمة الإزالة (DROP)

كونهـــا هـى الوحيدة القادرة على إزالة هيـــاكل البيانات، وبقية مكونات قاعدة البيانات (مثل القيود، والمنظورات)، ولإزالة هيكل الجدول المؤقت، على ســـبيل المثال، تستخدم التعليمة التالية:

```
DROP TABLE TEMP_T;
```

وعند اســـتخدام تعليمة الحذف، تجدر ملاحظة قيود الســـلامة المرجعية حيث إن حذف سجل ما قد يؤدى إلى حذف سجلات فى جداول أخرى مرتبطة بالسجل موضع الحذف بقيود المفتاح الخارجى، وحســـب تعريف القيود الخارجية، كما أسلفنا سابقاً، إما أن تقبل عملية الحذف ويتخذ الفعل المناســـب إزاء المفاتيح الخارجية هذه، (سواء وضع قيمها مســـاوية للقيمة غيـــر المعرفة أو لقيمة افتراضية مـــا)، أو ترفض عملية الحذف (عند ارتباط أحـــد المفاتيح الخارجية برد الفعل (RESTRICT))، أو يتم حذف الســـجل وجميع الســـجلات التى تشـــير إليه ضمن مفاتيحها الخارجية (عندما تكون مرتبطة برد الفعل (CASCADE)).

٨-٣-٣ تعليمة التحديث:

تســـتخدم تعليمة التحديث (UPDATE) لتحديث قيم حقول سجل واحد أو أكثر فى جدول ما، وكما هو الحال فى تعليمة الحذف الســـابقة، تســـتخدم العبارة الشـــرطية (WHERE) لتحديد الســـجلات التى ســـتجرى عليها عملية التحديث، فمثلاً، لتحديث راتب عضـــو هيئة التدريس ذى الرقم «٢٠٠» ليصبح ٤٠،٠٠٠ (عوضاً عن ٣٥،٠٠٠)، يمكن استخدام تعليمة التحديث كما يلى:

```
UPDATE FACULTY_T
SET SALARY = 40000
 WHERE FACULTY_ID ='200';
```

ويمكن اســـتخدام التعليمة لتحديث مجموعة من السجلات، عوضاً عن سجل واحد بحيث ينطبق عليها شـــروط عبارة (WHERE)، فمثلاً، يمكن اســـتخدام التعليمة التالية لزيادة مرتبات أعضاء هيئة التدريس الذين يعملون فى قسم الرياضيات بنسبة (١٠٪):

```
UPDATE FACULTY_T
SET SALARY = SALARY * 1.1
 WHERE DEPARTMENT_ID ='MATH';
```

كما يمكن تحديث السـجلات كافة فى الجدول، وذلك عند عدم استخدامنا للعبارة الشـرطية (WHERE) ضمن تعليمة التحديث، فمثلاً، يمكن زيادة رواتب جميع أعضاء هيئة التدريس فى الجامعة الأهلية بنسبة (١٠٪) وفق تعليمة التحديث التالية:

```
UPDATE FACULTY_T
SET SALARY = SALARY * 1.1;
```

وكمـا هو الحال عند اسـتخدام تعليمـة الحذف، تجدر ملاحظة قيود السـلامة المرجعيـة فى تعليمة التحديث حيث إن تحديث قيمة المفتاح الرئيسـى لسـجل ما قد يترتـب عليه تحديث المفاتيح الخارجية المعرفة فى جداول أخرى تشـير إلى السـجل موضع التحديث، فى هذه الحالة تتحدد عملية التحديث على السـجل من عدمها وفق الضوابـط الموضوعة على المفاتيح الخارجية والمصاحبـة لعبارة «عند التحديث» (ON UPDATE) فى الجداول التى تشير إلى السجل ضمن مفاتيحها الخارجية، كما أسلفنا سابقاً، فعلى سـبيل المثال، لو حاولنا تغيير المفتاح الرئيسى لعضو هيئة التدريس رقم «٢٠٠» ليصبح «٢٠٥»، فإن هذه العملية ستفشل، بمعنى أن نظام إدارة قاعدة البيانات لن يقوم بتنفيذها، والسبب وراء ذلك يعود إلى كون بعض السجلات المدونة فى جدول المجموعات الدراسية ترتبط بمفاتيح خارجية تشير إلى رقم عضو هيئة التدريس هذا على أساس أنه الشخص الذى يقوم بتدريس هذه المجموعات الدراسية، وأننا لم نربط المفتـاح الخارجى فى جدول المجموعات بعبـارة «عند التحديث» (On Update)، ومن ثم فإن الوضع الافتراضى عند التحديث هو التقييد (RESTRICT)، أى عدم التحديث.

```
UPDATE FACULTY_T
SET FACULTY_ID = '205'
WHERE FACULTY_ID = '200';
```

ويمكـن اسـتخدام تعليمة التحديث لتحديـث أكثر من حقل (وفى سـجل أو أكثر من السـجلات)، فمثلاً، يمكن اسـتخدام التعليمة التالية لزيادة مرتبات أعضاء هيئة التدريس الذين يعملون فى قسم الحاسب الآلى ("CS") بنسبة (١٠٪) ونقلهم إلى العمل فى قسم الرياضيات ("MATH").

```
UPDATE FACULTY_T
SET SALARY = SALARY * 1.1, DEPARTMENT_ID = 'MATH'
WHERE DEPARTMENT_ID = 'CS';
```

وتكون نتيجة التعليمة السابقة عند تنفيذها فى بيئة أوراكل، كما يلى:

```
SQL> UPDATE FACULTY_T
  2  SET SALARY = SALARY * 1.1, DEPARTMENT_ID = 'MATH'
  3  WHERE DEPARTMENT_ID = 'CS';

4 rows updated.
```

وباستعراض جدول أعضاء هيئة التدريس نجد أن رواتب وأقسام الأعضاء الأربعة الذين يعملون فى قسم الحاسب الآلى، وهم ذوو الأرقام الوظيفية (٣١٠، ٣٢٠، ٣٣٠، ٣٤٠) قد تم تحديثها بحيث تمت زيادة رواتبهم بنسبة (١٠٪) وأصبح القسم الذين يتبعونه هو قسم الرياضيات ("MATH") عوضاً عن قسم الحاسب الآلى "CS"، كما يلى:

FACULTY_	FNAME	LNAME	PHONE_NO	SALARY	DOB	DEPART
200	Khalid	Aloufi	454-2341	35000	22-MAY-63	MATH
220	Fahad	Alhamid	456-7733	25900	07-OCT-70	MATH
310	Saleh	Aleesa	454-8932	33000	13-SEP-66	MATH
320	Mohammed	Alhamad	454-5412	48400	13-MAY-65	MATH
330	Ghanim	Alghanim	456-2234	48950	12-AUG-69	MATH
340	Ibraheem	Alsaleh	454-1234	27500	20-JAN-70	MATH
400	Ahmad	Alotaibi	454-4563	33900	17-MAY-71	CHEM
420	Saleh	Alghamdi	454-2233	44600	13-FEB-69	CHEM
500	Yahya	Khorshid	456-2221	36700	12-MAR-65	ENGL
540	Salem	Alhamad	456-3304	40000	11-SEP-72	ENGL
560	Salman	Albassam	454-7865	33800	13-SEP-68	ENGL
600	Turki	Alturki	456-7891	27800	23-JUL-75	STAT
640	Fahad	Alzaid	456-3322	44300	12-MAY-71	STAT
660	Saud	Alkhalifa	454-9856	44900	13-AUG-72	STAT
710	Mahmood	Alsalem	456-3323	31900	19-FEB-73	PHYS
730	Mishal	Almazid	454-2343	29800	17-SEP-75	PHYS
770	Sultan	Aljasir	456-3212	43300	13-MAY-70	PHYS
800	Ali	Albader	456-7812	45300	22-JUN-66	EE
810	Saad	Alzhrani	454-5578	44200	17-OCT-67	EE
850	Ahmad	Alsabti	456-0120	33900	15-APR-73	EE

٨-٤ دوال الوقت والتاريخ، ودوال الأرقام، ودوال السلاسل الحرفية، ودوال التحويل:

توفر لغة الاستفسار البنائية مجموعة من الدوال التى تمكن من معالجة البيانات المخزنة فى قاعدة البيانات. ولاستكمال شرح تعليمات لغة الاستفسار البنائية فى بيئة أوراكل نشرح فى هذا الجزء بعضاً من هذه الدوال. وتسمى هذه الدوال فى بعض الأحيان بدوال الصفوف (Row Functions)، وذلك للتفريق بينها وبين دوال التجميع (التى تسمى أحياناً دوال الأعمدة)، وسبق أن قمنا بشرحها فى الجزء (٧-٢-١-٨).

٨-٤-١ دوال الوقت والتاريخ:

تخـــزن بيانات الوقت والتاريخ فـــى بيئة أوراكل بوصفها بيانـــات رقمية لتمثيل ما يلى:

القرن	CENTURY
السنة	YEAR
الشهر	MONTH
اليوم	DAY
الساعات	HOURS
الدقائق	MINUTES
الثوانى	SECONDS

والصيغـــة الضمنيـــة لإدخال وعرض التاريخ هـــى (DD-MON-YY) بحيث إن (DD) تمثل تاريخ اليوم، و(MON) تمثل الثلاثة أحرف الأولى من الشـــهر، و(YY) تمثل السنة، كما فى (20-JUL-99). وتســـتخدم الدالة (SYSDATE) لاسترجاع تاريخ اليوم من نظام التشـــغيل، وذلك باستخدام جدول افتراضى (Dummy) مخصص لهذا الغرض فى بيئة أوراكل هو (SYS.DUAL). فعلى سبيل المثال يمكن استعراض تاريخ اليوم، كما يلى:

```
SELECT SYSDATE
FROM SYS.DUAL;
```

وتكون نتيجة التعليمة السابقة، كما يلى:

```
SYSDATE
---------
16-JUN-07
```

ويمكـــن إجراء عمليات مختلفة علـــى الوقت والتاريخ من ضمنهـــا جمع عدد على تاريخ، وطرح عدد من تاريخ، وطرح تاريخ من تاريخ، كما يلى:

لحساب تاريخ الغد: (SYSDATE + 1)

لحساب تاريخ الأمس: (SYSDATE – 1)

لحساب الوقت بعد ست ساعات: (SYSDATE + 6/24)، بحيث إن عدد ساعات اليوم هو «٢٤» ساعة.

لحســـاب الوقت بعد عشر دقائق: (SYSDATE + 10/1440)، بحيث إن عدد الدقائق فى اليوم هو «١٤٤٠» دقيقة.

لحســاب الوقت بعد عشــر ثوان: (SYSDATE + 10/86400)، بحيث إن عدد الثوانى فى اليوم هو «٨٦٤٠٠» ثانية.

كما يمكن استخدام الدوال التالية للتعامل مع الوقت والتاريخ:

١- لإضافة أو طرح عدد (n) من الشهور من تاريخ (date) طبقاً لإشارة ((±) n) تستخدم الدالة التالية:

```
ADD_MONTHS (date, n)
```

٢- لإيجاد فرق الشــهور بين تاريخين بحيث يكون الناتج سالباً إذا كان التاريخ (date1) أصغــر مــن (date2)، كما قد يحتوى الناتج على جزء عشــرى يمثل فرق الأيام بين التاريخين، تستخدم الدالة التالية:

```
MONTHS_BETWEEN (date1, date2)
```

٣- لتقريب التاريخ والوقت طبقاً لشكل (Format) معين ويكون التقريب إلى أقرب سنة، أو شهر، أو أى جزء من أجزاء التاريخ والوقت، ومع إهمال (Format) يكون التقريب إلى منتصف ليل أقرب يوم، تستخدم الدالة التالية:

```
ROUND (date[,format])
```

٤- لاستقطاع جزء من التاريخ والوقت طبقاً لشكل (Format) معين، ومع إهمال (Format) يكون الوقت هو الصفر (أى منتصف الليل) (12.00AM)، تستخدم الدالة التالية:

```
TRUNC (date[,format])
```

٥- لإيجــاد تاريخ آخر يوم من الشــهر الذى يقع فيه التاريخ (date)، تســتخدم الدالة التالية:

```
LAST_DAY (date)
```

فعلى ســبيل المثال، لمعرفة تاريخ آخر يوم من شــهر مارس لعام ١٩٩٩م، تســتخدم التعليمة التالية:

```
SELECT LAST_DAY ('05-MAR-99')
FROM DUAL;
```

وتكون نتيجة التعليمة السابقة، كما يلى:

```
LAST_DAY(
---------
31-MAR-99
```

٨-٤-٢ دوال الأرقام:

توفـر لغة الاستفسـار البنائية فى بيئـة أوراكل مجموعة من الـدوال التى يمكن استخدامها مع الأرقام. وهذه الدوال كما يلى:

١- لتقريب حقل رقمى (A1) إلى حقل رقمى يحتوى على (A2) خانة على يمين الفاصلة العشرية، وبحيث يكون الناتج عددًا صحيحاً إذا كانت (0 = A2).

```
ROUND (A1,[A2])
```

ومن استخدامات الدالة السابقة خفض عدد الأرقام العشرية الناتجة من استخدام دوال الأعمدة (أو التجميع) فى جدول النتائج.

٢- لتمثيـل حقـل رقمى (A1) يحتوى علـى (A2) خانة على يمين الفاصلة العشـرية، تستخدم الدالة التالية:

```
TRUNC (A1,[A2])
```

٣- لإرجاع باقى قسمة العدد (A1) على العدد (A2)، تستخدم الدالة التالية:

```
MOD (A1,A2)
```

٤- لإرجـاع القيمة المطلقة للعدد (A1)، مـع ملاحظة أن القيمة المطلقة دائماً موجبة، تستخدم الدالة التالية:

```
ABS (A1)
```

٥- لمعرفة إشـارة العدد (A1) بحيث تكون النتيجة هى (1) إذا كان A1 موجباً و(1-) إن كان سالباً و(0) إذا كان العدد يساوى صفراً، تستخدم الدالة التالية:

```
SIGN (A1)
```

فعلى سبيل المثال، لمعرفة متوسط مرتبات أعضاء هيئة التدريس الذين يعملون فى قسم الفيزياء ومتوسط مرتبات أعضاء هيئة التدريس الذين يعملون فى قسم الكيمياء مقربين إلى رقمين عشريين وإلى أقرب عددين صحيحين، نستخدم التعليمة التالية:

```
SELECT DEPARTMENT_ID, AVG(SALARY), ROUND(AVG(SALARY),2),
TRUNC(AVG(SALARY))
FROM FACULTY_T
WHERE DEPARTMENT_ID = 'PHYS' OR DEPARTMENT_ID = 'CHEM'
GROUP BY DEPARTMENT_ID;
```

وتكون نتيجة التعليمة السابقة، كما يلى:

```
DEPART AVG(SALARY) ROUND(AVG(SALARY),2) TRUNC(AVG(SALARY))
------ ----------- -------------------- ------------------
CHEM         39250                39250              39250
PHYS         35000                35000              35000
```

٨-٤-٣ دوال السلاسل الحرفية:

١- لاختيار جزء من سلسلة حرفية (A1) ابتداء من الموقع (A2) وبحيث تمثل (A3) عدد الحروف المطلوبة فى السلسلة الحرفية الناتجة، مع ملاحظة أنه عند حذف (A3) تحتوى النتيجة على كل حروف (A1) التى على يمين (A2)، تستخدم الدالة التالى:

```
SUBSTR (A1,A2[,A3])
```

فعلى سبيل المثال، تكون نتيجة التعليمة التالية الحروف الأربعة الأولى من السلسلة الحرفية ابتداءً من أول حرف فيها:

```
SELECT SUBSTR ('1994-01-01',1,4)
FROM DUAL;
```

وتكون نتيجة التعليمة السابقة، كما يلى:

```
SUBS
----
1994
```

٢- لإيجـاد طول الحقل (A1) متضمنـاً ذلك الفراغات والأصفار السـابقة (Leading Zeros)، تستخدم الدالة التالية:

```
LENGTH (A1)
```

٣- لربط سلسلتين حرفيتين قد تكونان قيماً فى عمودين أو قيمة عمود وثابت حرفى كعمود واحد، تستخدم الدالة التالية:

```
A1 || A2
```

٤- لتحويل جميع حروف سلسلة حرفية (إنجليزية) إلى حروف كبيرة، تستخدم الدالة التالية:

```
UPPER (A1)
```

٥- لتحويـل جميع حروف سلسـلة حرفية (إنجليزية) إلى حروف صغيرة، تسـتخدم الدالة التالية:

```
LOWER(A1)
```

٦- لتحويل أول حرف من كل كلمة (إنجليزية) فى سلسلة حرفية إلى حرف كبير بحيث تكون الفواصل بين الكلمات هى المسافة (SPACE) أو أحد الرموز التالية (.: ؛ # ؛ $ أو غيرها)، تستخدم الدالة التالية:

```
INITCAP(A1)
```

٨-٤-٤ دوال التحويل:

١- تتعامل البرامج مع الشـكل الخارجى للتاريخ سلسـلة حرفية، ويتم تحويله مباشرة إلى الصيغة الضمنية. وتسـتخدم الدالة التالية لتحويل تاريخ (date) إلى سلسـلة حرفية طبقاً لشكل (Format) معين:

```
TO_CHAR(date, format)
```

وفيما يلى بعض الأشكال القياسية:

DATE	TIME	FORMAT
yyyy-mm-dd	hh.mm.ss	ISO
mm/dd/yyyy	Hh:mm PM or hh:mm AM	USA
dd.mm.yyyy	hh.mm.ss	EUR

وبإهمال الشــكل (Format)، يتم تحويل التاريخ طبقاً للصيغة الضمنية. وفيما يلى الأشكال (Format) المختلفة لصيغة التاريخ:

DD	رقم اليوم من الشهر (١ إلى ٣١)
DAY	لعرض اسم اليوم كاملاً (Sunday to Friday) فى ٩ خانات
DY	لعرض اسم اليوم مختصراً (Sun to Fri)
MM	رقم الشهر من السنة (١ إلى ١٢)
MONTH	لعرض اسم الشهر كاملاً (January to December) فى ٩ خانات
MON	لعرض اسم الشهر مختصراً (Jan to Dec)
YY	أول رقمين من السنة، ٩٨ مثلاً
YYYY	رقم السنة كاملاً، ١٩٩٨ مثلاً
CC	رقم القرن الميلادى
HH or HH12	الساعة من ١ إلى ١٢
AM or PM	لتحديد الوقت ما إذا كان قبل أو بعد منتصف الليل
HH24	الساعة من ١ إلى ٢٤
MI	الدقيقة من ١ إلى ٦٠
SS	الثانية من ١ إلى ٦٠
; : - / ,	علامات التوقيف
"text"	نص داخل علامات تنصيص
TH	رتبة الأرقام كما فى (1st, 2nd, 3rd, 4th, 5th, ...)
SP	الأرقام كتابة (first, second, ...)
FM	أسماء الأيام والشهور دون إضافة فراغات (Blank Padding)

ومن الأمثلة التطبيقية على تحويل التاريخ إلى سلاسل حرفية ما يلى:

طريقة التحويل	النتيجة
TO_CHAR(SYSDATE, 'fmMonth, ddth,yyyy')	May, 12th, 1998
TO_CHAR(SYSDATE, 'Month, ddsp,yyyy')	May , twelve, 1998
TO_CHAR(SYSDATE, '"On the" ddspth "of" fmMONTH "at" hh:mi:ssPM')	On the Twelfth of MAY at 11:34:29AM

٢- لتحويل الرقم (Number) إلى سلسلة حرفية، تستخدم التعليمة التالية:

```
TO_CHAR (Number[,format])
```

بحيث يمكن أن يكون الشكل (Format) على إحدى الصيغ التالية:

٩٩٩٩٠	عدد التسعات والأصفار يحدد عدد الخانات الممكن عرضها
999,999.99	يمكن استخدام الفاصلة والفاصلة العشرية للتحكم فى طريقة العرض
$999	لعرض الرقم كعملة.
S999	لعرض الإشارة (- أو +) قبل الرقم
999S	لعرض الإشارة (- أو +) بعد الرقم
999MI	لعرض (-) بعد الرقم إذا كان الرقم سالباً. لا تظهر الإشارة الموجبة
RN	لعرض الرقم بشكل الأرقام الرومانية

٣- لتحويل سلسلة حرفية إلى تاريخ طبقاً للشكل (Format)، وبإهمال (Format) يجب أن تكون السلسلة الحرفية مطابقة للصيغة الضمنية (dd-mon-yyyy)، يمكن استخدام جميع أشكال (Format) المستخدمة مع دالة (TO_CHAR) عدا ("text", th, sp, fm)، تستخدم الدالة التالية:

```
TO_DATE (string[,format])
```

ومن الأمثلة التطبيقية على تحويل السلاسل الحرفية إلى تواريخ ما يلى:

طريقة التحويل	النتيجة
TO_DATE('12-MAY-98')	12-May-98
TO_DATE('May, 12, 1998', 'Month, dd, yyyy')	12-May-98

٨-٥ لغة التحكم فى البيانات ((Data Control Language (DCL)):

توفر نظـم إدارة قواعد البيانات العلاقية إمكانية التحكم فى الصلاحيات المخولة للمســتفيدين للتعامـل مع البيانات المخزنة فى قاعدة البيانـات. ولأنه يمكن التعامل مع قاعدة البيانات الواحدة من قبل أكثر من مســتخدم وفى آن واحد، فإن نظم إدارة قواعد البيانات العلاقية تمكن من التعامل المتزامن للبيانات من قبل أكثر من مستخدم، وذلك عن طريق تزويد كل مســتخدم لنظام إدارة قاعــدة البيانات برمز خاص يمكنه من التعامل مع قاعدة البيانات. ويســتخدم الرمز الخاص بكل مســتخدم فى تعريف الصلاحيـات التى تخولـه للتعامل مع الجداول المختلفة المعرفـة فى قاعدة البيانات. وتوفر لغة الاستفسار البنائية تعليمتين يمكن من خلالهما التحكم فى الصلاحيات التى تعطى للمســتفيدين للتعامل مع قاعدة البيانات أو ســحب الصلاحيات منهم، وهاتان التعليمتان هما تعليمة منح الصلاحية (Grant) وتعليمة سحب الصلاحية (Revoke).

٨-٥-١ منح الصلاحيات:

عند إنشــاء جدول جديد باستخدام تعليمة الإنشــاء (Create) يكون الجدول المنشأ ملكاً للمستخدم (أو المستفيد) الذى قام بإنشائه. ولمعرفة ملاك الجداول المختلفة فى قاعدة البيانات تقوم نظم إدارة قواعد البيانات العلاقية (داخلياً) بوضع رمز المستخدم الذى يملك الجدول قبل اسم الجدول. فلو افترضنا أن رمز المستخدم الذى قام بإنشاء جــدول المواد الدراســية (Course_T) هو (Houmaily) فإن نظــام إدارة قاعدة البيانات ســيقوم بتسمية الجدول داخلياً بالاسم نفسه الذى استخدم فى تعليمة إنشاء الجدول مسبوقاً باسم المستخدم الذى قام بإنشائه، كما يلى:

```
HOUMAILY.COURSE_T
```

وتعنى الطريقة الســابقة فى تســمية الجداول داخل قاعدة البيانات أنه بالإمكان إنشــاء أكثر من جدول بالمســمى نفســه، ولكن من قبل مســتخدمين (أو مستفيدين) مختلفــين. ويقــوم نظام إدارة قواعــد البيانات فى حل أى التباس قــد يظهر نتيجة لتكرار مســميات الجداول من خلال إدراج رمز المستفيد الذى يملك الجدول قبل اسم الجــدول، كما هو أعلاه. وعند الرجوع إلى جدول يتكرر اســمه ضمن جداول قاعدة البيانات فإنه يتعين على المســتفيد إدراج رمز مالك الجدول قبل اســم الجدول حتى يتمكــن نظام إدارة قاعدة البيانات مــن التعرف على الجدول المقصود دون أى التباس

مع الجداول الأخرى التى تحمل المسمى نفسه. وفى حالة عدم التقيد بذلك من قبل المستفيد فإن نظام إدارة قاعدة البيانات لن يتمكن من التعرف على الجدول المقصود، ومن ثم فإنه لن يقوم بتنفيذ العملية الصادرة من قبل المستفيد التى تحتوى على اسم جدول يتكرر مع مسميات جداول أخرى فى قاعدة البيانات. أما فى حالة عدم وجود التباس فى مسميات الجداول، فإنه يمكن استخدام اسم الجدول مباشرة دون إدراج رمز المستخدم الذى يملك الجدول.

وباستطاعة المستفيد الذى قام بإنشاء جدول ما التعامل مع هيكل الجدول والبيانات المخزنة فيه، إذ إن بإمكانه حذف هيكل الجدول أو التعديل عليه، كما أن بإمكانه استرجاع البيانات الموجودة فى الجدول والتعديل عليها (من خلال عمليات الحذف والإضافة والتحديث). ولأنه بالإمكان منح صلاحيات محددة على أى جدول لمستفيدين آخرين خلاف الشخص المالك للجدول الذى يملك الصلاحيات الكاملة للتعامل مع هيكل الجدول ومحتوياته، فإن نظام إدارة قاعدة البيانات يقوم بتحديد الصلاحيات الممنوحة لكل مستخدم وتدوينها فى ملفات خاصة بالنظام. وتمكن نظم إدارة قواعد البيانات العلاقية من إعطاء الصلاحيات التالية للمستفيدين على جداول قاعدة البيانات:

صلاحية الاسترجاع (Select Privilege): تمكن هذه الصلاحية من استرجاع البيانات المخزنة فى الجدول باستخدام تعليمة الاختيار (أو الاسترجاع) (Select Statement).

صلاحية الإضافة (Insert Privilege): تمكن هذه الصلاحية من إضافة سجلات جديدة للجدول باستخدام تعليمة الإضافة (Insert Statement).

صلاحية الحذف (Delete Privilege): تمكن هذه الصلاحية من حذف سجلات موجودة فى الجدول باستخدام تعليمة الحذف (Delete Statement).

صلاحية التحديث (Update Privilege): تمكن هذه الصلاحية من تعديل القيم المخزنة فى سجلات الجدول باستخدام تعليمة التحديث (Update Statement).

وعند إنشاء جدول جديد يكون للمستفيد الذى قام بإنشاء الجدول وحده كافة الصلاحيات المدرجة أعلاه، فى حين لا يمتلك أى مستفيد آخر أى صلاحية للتعامل مع محتويات الجدول. وحتى يتمكن مستفيد آخر من التعامل مع محتويات الجدول فإنه لا بد أن يقوم مالك الجدول بمنح المستفيد بعض الصلاحيات المدرجة أعلاه.

ولمنح المستفيد صلاحية التعامل مع الجدول تستخدم تعليمة (Grant) التى تأخذ الشكل العام التالى:

```
GRANT Privileges ON Table_Name TO User;
```

ويقصد فى (Privileges) الصلاحيات الممنوحة، و(Table_Name) اسم الجدول الذى ستمنح عليه الصلاحيات، و(User) رمز المستفيد الذى سيتم منحه للصلاحيات. فعلى سبيل المثال، يمكن منح صلاحية الاسترجاع للمستفيد (Student1) على جدول المواد الدراسية من قبل مالك الجدول، وليكن المستفيد (Houmaily)، كما يلى:

```
GRANT SELECT ON COURSE_T TO STUDENT1;
```

ويمكن أيضاً منح أكثر من صلاحية، من خلال تعليمة منح الصلاحية نفسها، للتعامل مع الجدول من قبل المستفيد، كما يلى:

```
GRANT SELECT, UPDATE, DELETE ON COURSE_T TO STUDENT1;
```

وباستطاعة المستفيد (Student1) الآن تنفيذ تعليمات الاسترجاع والتحديث والحذف على جدول المواد الدراسية. وتوفر لغة الاستفسار البنائية إمكانية منح الصلاحيات كافة المدرجة أعلاه بشكل مختصر دون الحاجة إلى سرد الصلاحيات الواحدة تلو الأخرى ضمن تعليمة منح الصلاحية وذلك من خلال استخدام عبارة (All Privileges)، كما يوضح المثال التالى:

```
GRANT ALL PRIVILEGES ON COURSE_T TO STUDENT1;
```

ويمكن أيضاً الاستغناء عن الكلمة الاختيارية (Privileges) من التعليمة أعلاه لتصبح أكثر اختصاراً، كما يلى:

```
GRANT ALL ON COURSE_T TO STUDENT1;
```

وبعد تنفيذ التعليمة أعلاه يصبح للمستفيد (Student1) الصلاحيات كافة التى يملكها المستفيد الذى قام بإنشاء الجدول، والتى تمكنه من التعامل مع محتويات الجدول من خلال أية تعليمة من تعليمات لغة معالجة البيانات. ويمكن أيضاً منح الصلاحيات لأكثر من مستخدم فى الوقت نفسه. فعلى سبيل المثال، يمكن منح صلاحية استرجاع

بيانات جدول المواد الدراسـية لكل من (Student2) و(Student3) و(Student4)، من خلال استخدام تعليمة منح الصلاحية نفسها كما يلى:

```
GRANT SELECT ON COURSE_T TO STUDENT2, STUDENT3, STUDENT4;
```

ولإعطاء صلاحية معينة على جدول ما لجميع المستفيدين من قاعدة البيانات عوضاً عن مستفيدين محددين تستخدم كلمة (Public)، أى لعموم المستفيدين. فعلى سبيل المثال، يمكن منح صلاحية استرجاع بيانات جدول المواد الدراسية لعموم المستفيدين، كما يلى:

```
GRANT SELECT ON COURSE_T TO PUBLIC;
```

وتعنى التعليمة السابقة أنه بإمكان أى مستفيد أن يقوم بتنفيذ تعليمة الاسترجاع على جدول المواد الدراسية. كما يمكن استخدام كلمة (ALL) مع كلمة (PUBLIC) لمنح الصلاحيات كافة على جدول ما، ولجميع المستفيدين من قاعدة البيانات. فعلى سبيل المثال، يمكن منح جميع الصلاحيات لجميع المستفيدين على جدول المواد الدراسية، كما يلى:

```
GRANT ALL ON COURSE_T TO PUBLIC;
```

وتوفر تعليمة منح الصلاحية إمكانية إعطاء الصلاحيات على حقول معينة فى الجدول عوضاً عن جميع حقوله. وبهذه الطريقة يمكن حجب التعامل مع بعض البيانات الحساسة فى الجدول عن المستفيدين مما يقدم حماية أكثر دقة للبيانات المخزنة فى الجدول. فعلى سبيل المثال، يمكن إعطاء صلاحية تحديث القسم الدراسى الذى تتبعه المادة الدراسية لجميع المستفيدين، مع حجب إمكانية تحديث أى من الحقول الأخرى فى جدول المواد الدراسية، كما يلى:

```
GRANT UPDATE(DEPARTMENT_ID) ON COURSE_T TO PUBLIC;
```

وتجدر الإشارة إلى أن مقياس لغة الاستفسار البنائية ينص على أنه بالإمكان تحديد الحقول التى بالإمكان منح الصلاحية عليها عندما تكون الصلاحية المعطاة هى صلاحية التحديث (Update). أما بالنسبة للصلاحيات الأخرى فإن منح الصلاحية سيكون على حقول الجدول كافة، إلا أن بعض نظم إدارة قواعد البيانات العلاقية تمكن من منح الصلاحيات الأخرى (غير صلاحية التحديث) على أعمدة محددة عوضاً عن حقول الجدول كافة.

٨-٥-١-١ منح الصلاحيات على المنظورات:

يوجد للمنظورات، شـأنها شأن الجداول، صلاحيات يمكن منحها وكذلك حجبها. وكما هو الحال بالنسبة للجداول يمكن منح الصلاحيات باستخدام تعليمة (Grant). فعلى سبيل المثال، يمكن منح صلاحية الاسترجاع لجميع المستفيدين على منظور جدول المواد الدراسية، على افتراض وجود مثل هذا المنظور ضمن هياكل قاعدة البيانات، كما يلى:

```
GRANT  SELECT ON COURSE_V TO PUBLIC;
```

إلا أنه من الضرورة بمكان الإشارة إلى أن منح صلاحية التحديث، والإضافة، والحذف تعد أكثر تعقيداً؛ وذلك لأن المنظور قد لا يكون قابلاً للتحديث عليه، أو الإضافة إليه، أو الحذف منه كما سبق أن أوضحنا عند حديثنا عن المنظورات. وتعزى هذه المعضلة إلى تعريف المنظور فى أثناء إنشائه باستخدام تعليمة إنشاء المنظور (... Create View). وبناءً على تعريف المنظور فإن بعض عمليات منح الصلاحية قد لا يمكن تنفيذها لكونها تتضارب مع العمليات التى يمكن تنفيذها على المنظور نفسه. فعلى سبيل المثال، لا يمكن منح صلاحية التحديث على منظور إذا كان معرفاً بطريقة لا تقبل إجراء عمليات التحديث عليه، مثل احتوائه على دوال تقوم بتجميع البيانات (Aggregate Functions) أو إن كان يدمج بين محتويات أكثر من جدول.

٨-٥-١-٢ إعطاء الحق فى تخويل الصلاحية:

تسمح لغة الاستفسار البنائية لمالك الجدول بإعطاء حق ممارسة تخويل الصلاحية لمستفيد (أو مجموعة من المستفيدين)، بمعنى أن يصبح لهذا المستفيد (أو مجموعة المستفيدين) القدرة على منح الصلاحيات التى خولت لهم من قبل مالك الجدول لمستفيدين آخرين. ويمكن ممارسة هذا الحق من خلال استخدام عبارة (With Grant Option). فعلى سبيل المثال، يمكن للمستفيد (Houmaily) منح الصلاحيات كافة التى يملكها على جدول المواد الدراسية، الذى قام بإنشائه، للمستفيد (Student1) مع إعطائه الحق فى تخويل الصلاحيات التى أعطيت له لأى مستفيد آخر، كما يلى:

```
GRANT  ALL ON COURSE_T TO STUDENT1
WITH GRANT OPTION;
```

وتعنى عبارة (With Grant Option) الواردة فى نهاية التعليمة السابقة أن مالك الجدول وهو (Houmaily) قد أعطى المستفيد (Student1) الحق فى تخويل الصلاحيات الممنوحة له لمستفيدين آخرين بالإضافة إلى حقه فى ممارسة الصلاحيات المعطاة له للتعامل مع محتويات الجدول. وبناءً على حق تخويل الصلاحية الذى منح للمستفيد (Student1) من قبل مالك الجدول فإنه بإمكان المستفيد (Student1) تخويل بعض أو كل الصلاحيات التى تم إعطاؤها له على الجدول لمستفيدين آخرين. ويعنى هذا أنه من ضمن الصلاحيات التى أعطيت للمستفيد (Student1) صلاحية استخدام تعليمة منح الصلاحية (Grant) على جدول المواد الدراسية بما يتوافق مع صلاحيات الاسترجاع والتعديل التى أعطيت له. وتجدر الإشارة إلى أنه ليس من الضرورى أن تعطى جميع الصلاحيات لمستفيد ما مع صلاحية الحق فى تخويل الصلاحية حتى يتمكن من ممارسة حقه فى تخويل الصلاحية، كما هو الحال فى المثال السابق. فعلى سبيل المثال، يمكن منح المستفيد (Student1) الحق فى تنفيذ عمليات الاسترجاع فقط على جدول المواد الدراسية مع حق تخويل الصلاحية. وفى هذه الحالة يمكن للمستفيد (Student1) تنفيذ تعليمات الاسترجاع على جدول المواد الدراسية، وكذلك تخويل هذه الصلاحية فقط لمستفيدين آخرين.

٨-٥-٢ سحب الصلاحيات:

ولأنه بالإمكان منح الصلاحيات للمستفيدين فإنه بالإمكان كذلك سحب الصلاحيات منهم. وتستخدم تعليمة سحب الصلاحيات (Revoke) لسحب الصلاحيات الممنوحة للمستفيدين. وتعمل تعليمة سحب الصلاحية على سحب صلاحيات محددة من المستفيدين، مثلها مثل تعليمة منح الصلاحية التى تعطى صلاحيات محددة للمستفيدين. فعلى سبيل المثال، يمكن سحب صلاحية التحديث من المستفيد (Student1) الممنوحة له على جدول المواد الدراسية، كما يلى:

```
REVOKE UPDATE ON COURSE_T FROM STUDENT1;
```

أما إذا ما أريد سحب صلاحية الإضافة وصلاحية الحذف من المستفيد (Student1) الممنوحتين له على جدول المواد الدراسية، فإنه يمكن تنفيذ تعليمة سحب الصلاحية التالية:

```
REVOKE INSERT, DELETE ON COURSE_T FROM STUDENT1;
```

كما يمكن أن تستخدم عبارة (ALL) ضمن تعليمة سحب الصلاحية كاختصار يقصد به جميع الصلاحيات الممنوحة. فعلى سـبيل المثال، يمكن سـحب جميع الصلاحيات الممنوحة للمستفيد (Student1) باستخدام عبارة (ALL)، كما يلى:

```
REVOKE  ALL ON COURSE_T FROM STUDENT1;
```

وكذلك يمكن استخدام عبارة (Public) كاختصار يُقْصَد به المستفيدون كافةً. فعلى سـبيل المثال، يمكن سـحب صلاحية الإضافة وصلاحية الحذف وصلاحية التحديث من المستفيدين كافة، كما يلى:

```
REVOKE  INSERT, DELETE, UPDATE ON COURSE_T FROM PUBLIC;
```

أما إذا أريد حجب إمكانية التعامل مع جدول المواد الدراسية عن المستفيدين كافة فإنه يمكن تنفيذ تعليمة سحب الصلاحية التالية:

```
REVOKE ALL ON COURSE_T FROM PUBLIC;
```



الفصل التاسع

موضوعات متقدمة فى نظم قواعد البيانات

يتطرق هذا الفصل، باقتضاب، إلى أربعة موضوعات متطورة ومهمة فى نظم قواعد البيانات وهى: المعاملات، وقواعد البيانات الشيئية، وقواعد البيانات العلاقية-الشيئية، وقواعد البيانات الموزعة. تمثل المعاملات الوسيلة الرئيسية التى يتم من خلالها التفاعل مع قواعد البيانات من قبل المستفيدين، سواء بشكل تفاعلى مباشر أم من خلال برامج التطبيقات التى يقوم مطورو التطبيقات ببنائها. أما نموذج البيانات الشيئى فقد تم تطويره لسد الاحتياجات التقنية التى يتطلبها تطوير نظم التطبيقات المتعلقة بمكننة أعمال المنظمات ذات الصبغة غير التقليدية من حيث البيانات التى تتعامل معها مثل استخدامها فى تطبيقات التصميم بمساعدة الحاسب الآلى (Computer-Aided Design)، والتصنيع بمساعدة الحاسب الآلى (Computer-Aided Manufacturing)، والتجارب العلمية، ونظم المعلومات الجغرافية (Geographical Information Systems)، وتطبيقات الوسائط المتعددة (Multimedia Applications)؛ على سبيل المثال لا الحصر. ونظراً لانتشار النموذج العلاقى وسهولته فى تمثيل البيانات والتعامل معها، عكف الكثير من الشركات المصنعة لنظم إدارة قواعد البيانات العلاقية على تبنى بعض مفاهيم النموذج الشيئى ضمن منتجاتها حتى تتمكن من مواكبة احتياجات المنظمات التى تتصف بياناتها بالصبغة غير التقليدية بالإضافة إلى تلك التى تتصف بالتقليدية. وأصبحت مثل هذه المنتجات تسمى قواعد البيانات العلاقية-الشيئية. أما بالنسبة للمنظمات التى تتوزع فيها مقراتها فى مناطق عددية، وعلى رقع متباعدة جغرافياً فى الكثير من الأحيان، فقد دفعت هذه المنظمات الباحثين إلى تبنى مفهوم النظم الموزعة وأضحت تسمى فى مجال نظم قواعد البيانات «نظم قواعد البيانات الموزعة». وتوفر مثل هذه النظم العديد من الميزات مقارنة بتلك النظم المركزية من ضمنها «الموثوقية» (Reliability) و «التواجد» (Availability) هذا بالإضافة إلى أدائها المتميز وسهولة التوسع فى الأجهزة والتطبيقات فى مثل هذه المنظمات. ونظراً لأهمية المفاهيم الأربعة السابقة كان من الضرورى التطرق إليها فى هذا الكتاب ولو بشكل مقتضب.

٩-١ المعاملات (Transactions):

تعد المعاملات الوسيلة الرئيسية التى يتم من خلالها التفاعل مع قواعد البيانات من قبل المستفيدين سواء بشكل تفاعلى مباشر أم من خلال برامج التطبيقات التى يقوم مطورو التطبيقات ببنائها. وتُعرف المعاملات على أنها برنامج (أو جزء) من برنامج (حاسوبي) يتم من خلاله التفاعل مع قاعدة بيانات بحيث يقوم بتحويل قاعدة البيانات من حالة صحيحة إلى حالة أخرى صحيحة تتوافق مع الضوابط المفروضة على قاعدة البيانات. وقد يكون البرنامج الحاسوبى مكوناً بالكامل من تعليمات تتفاعل مع قاعدة البيانات، مثل تعليمات لغة الاستفسار البنائية، وذلك عندما يتم التفاعل مع قاعدة البيانات بشكل مباشر (أو تفاعلى) (Interactive Mode) دون تضمين هذه التعليمات فى برنامج مكتوب بلغة برمجة عامة (General Purpose Programming Language)؛ أو قد يكون البرنامج مكوناً من تعليمات تتفاعل مع قاعدة البيانات، ولكن هذه التعليمات مكتوبة ضمن ثنايا إحدى لغات البرمجة العامة (مثل سى، وجافا، وكوبول ... إلخ). وفى كلتا الحالتين يتكون أو يحتوى البرنامج على تعليمات يمكن فهمها ومعالجتها من قبل نظام إدارة قاعدة البيانات.

وتقوم كل معاملة إذا تم تنفيذها بشكل منفرد على قاعدة البيانات دون تداخل مع أية معاملات أخرى تحت التنفيذ على قاعدة البيانات نفسها ودون أية أعطال للنظام فى أثناء تنفيذ المعاملة بنقل قاعدة البيانات من حالة سليمة إلى حالة سليمة أخرى لا تحتوى على بيانات تخترق أياً من القيود المفروضة على قاعدة البيانات. فعلى سبيل المثال، قد يحتوى البرنامج على بعض من تعليمات لغة الاستفسار البنائية إذا كان البرنامج يتفاعل مع قاعدة بيانات علاقية. ويعد كل تنفيذ لمجمل مجموعة التعليمات الموجودة فى البرنامج معاملة واحدة. وتنحصر التعليمات التى تتكون منها أية معاملة بين عملية «بداية» (Begin) تشير إلى بداية تنفيذ معاملة جديدة على قاعدة البيانات، وعملية «نهاية» (End) تشير إلى انتهاء المعاملة. ويتم إدراج عدد من عمليات التعديل (الإضافة والحذف والتحديث) والاسترجاع على قاعدة البيانات بين تعليمتى البداية والنهاية. وتستخدم عمليتا البداية والنهاية من قبل نظام إدارة قاعدة البيانات للتعرف على بداية كل معاملة ونهايتها، فى حين تستخدم التعليمات الأخرى للتفاعل مع محتويات قاعدة البيانات. كما أن غالبية نظم إدارة قواعد البيانات تتعرف على بداية كل معاملة بشكل ضمنى، وذلك عند تنفيذ أول تعليمة تتفاعل مع قاعدة البيانات من قبل أحد المستخدمين أو التطبيقات، فى حين يتم التعرف على نهاية

المعاملـــة ضمنياً أيضاً من خلال تنفيذ تعليمة التثبيت (Commit) أو تعليمة الانســحاب (Rollback)، التى تسمى أيضاً تعليمة الإخفاق (Abort)، من قبل المعاملة. وعندما تكون نهاية المعاملة تعليمة تثبيت فإن هذا يعنى أن المستخدم (أو التطبيق) يرغب فى تثبيت جميع التعديلات التى أجريت من قبل المعاملة التى قام بتنفيذها على قاعدة البيانات. أما عندما تكون نهاية المعاملة تعليمة انسحاب (أو إخفاق) فإن هذا يعنى أن المستخدم (أو التطبيق) يرغب فى عدم تثبيت أى من التعديلات التى قامت المعاملة بتنفيذها على قاعــدة البيانات مما يعنى إرجاع جميع قيم البيانات التى تفاعلت معها المعاملة إلى ما كانــت عليه قبل تنفيذ المعاملة كما لو أنه لم يتم تنفيــذ المعاملة على محتويات قاعدة البيانـــات على الإطلاق. وتتصف المعاملات فـــى قواعد البيانات بأربع خصائص هى (Bernstein et al, 1987; Gray and Reuter, 1992):

١- **النوويــة (Atomicity)**: تعنـــى هـــذه الخاصية أن كل معاملة تنفـــذ باعتبارها وحدة منطقية واحدة غير قابلـــة للتجزئة؛ بحيث إن كافة العمليات التى تحتويها المعاملة إما أن يتم تنفيذها بالكامل على قاعدة البيانات وإما أن لا يتم تنفيذ أى منها على الإطلاق.

٢- **الصحــة (أو التوافــق) (Consistency)**: تعنى هذه الخاصية أن كل معاملة عبارة عن جزء من برنامج (حاسوبى) قد تمت كتابته بشكل صحيح يتوافق مع الضوابط التى وضعت على قاعدة البيانات بالإضافة إلى الضوابط المرعية فى المنظمة.

٣- **العزلــة (Isolation)**: تعنى هذه الخاصية أن كل معاملة يتم تنفيذها بشـــكل منعزل علـــى قاعـــدة البيانات دون أى تداخل مـــع المعاملات الأخرى التـــى قد تكون قيد التنفيذ بالتزامن معها على قاعدة البيانات نفسها.

٤- **الدوام (Durability)**: تعنى هذه الخاصية أن أى تعديلات تجرى على قاعدة البيانات من قبل المعاملات التى تنتهى وتثبت نتائجها على قاعدة البيانات سيســـتمر وجود نتائجها على قاعدة البيانات حتى لو تعطل النظام مستقبلاً.

وتُعْرف الخصائص الأربع الســـابقة للمعاملات بمســـمى خصائص «أَسِـــد» (ACID Properties) اختصـــاراً لهـــا بحيـــث يمثل هذا المســـمى الحروف الأولى من مســـميات الخصائـــص أعلاه. وعند بناء نظم التطبيقات تتم كتابة المعاملات بداخل إحدى لغات البرمجـــة العامـــة مثل ســـى (C) أو جافا (JAVA) أو كوبـــول (COBOL)، أو داخل إحدى لغـــات البرمجة المخصصة لتطوير نظم التطبيقات التى تقوم الشـــركات المصنعة لنظم إدارة قواعـــد البيانات بتوفيرها لتطوير التطبيقات على نظم قواعد البيانات التى تقوم

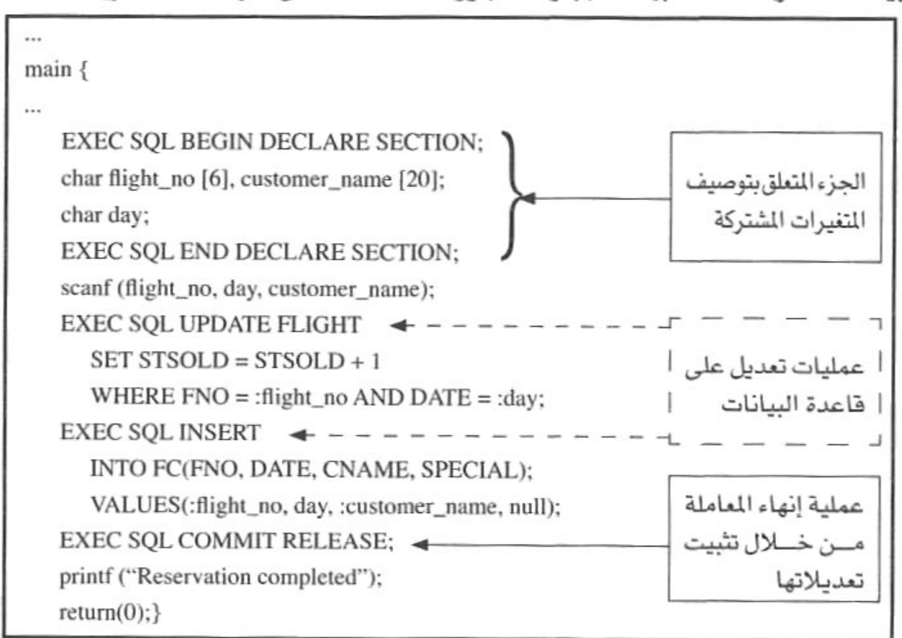
بتصنيعها مثل «أوراكل دفلوبر» (Oracle Developer) من شركة أوراكل. وتسمى اللغة التى تحتوى على تعليمات تتعامل مع قاعدة البيانات باللغة المضيفة (Host Language). فعلى سبيل المثال، لو افترضنا وجود قاعدة بيانات تتعلق بنظام حجز للرحلات الجوية مكونة من ثلاثة جداول، كما يلى:

١- جـدول الرحلات الجوية FLIGHT(<u>FNO, DATE</u>, SRC, DEST, STSOLD, CAP) الذى يتكـون من حقل رقـم الرحلة (FNO)، وتاريخهـا (DATE)، ومحطة الإقلاع (SRC)، ومحطـة الوصـول (DEST)، وعـدد المقاعد المباعـة (STSOLD)، وسـعة الطائرة (CAP).

٢- جـدول العملاء (أو الركاب) CUST(<u>CNAME</u>, ADDR, BAL) الذى يتكون من اسـم العميل (CNAME)، وعنوانه (ADDR)، ورصيده (BAL).

٣- جدول حجـوزات العمـلاء FC(<u>FNO, DATE, CNAME</u>, SPECIAL) الذى يتكون من حقل رقم الرحلة (FNO)، وتاريخها (DATE)، واسـم العميل (CNAME)، واحتياجات العميل الخاصة (SPECIAL).

فـإن البرنامـج التالى، المكتوب بلغة سـى (C)، يتضمن معاملـة تتفاعل مع قاعدة البيانات المعرفة أعلاه بهدف إجراء حجوزات للعملاء على الرحلات الجوية.



```
...
main {
...
    EXEC SQL BEGIN DECLARE SECTION;
    char flight_no [6], customer_name [20];
    char day;
    EXEC SQL END DECLARE SECTION;
    scanf (flight_no, day, customer_name);
    EXEC SQL UPDATE FLIGHT
        SET STSOLD = STSOLD + 1
        WHERE FNO = :flight_no AND DATE = :day;
    EXEC SQL INSERT
        INTO FC(FNO, DATE, CNAME, SPECIAL);
        VALUES(:flight_no, day, :customer_name, null);
    EXEC SQL COMMIT RELEASE;
    printf ("Reservation completed");
    return(0);}
```

ويتكون البرنامج السابق من جزء يتعلق بتعريف المتغيرات المشتركة التى تستخدمها كل من لغة البرمجة المضيفة ولغة الاستفسار البنائية. ويتم وضع هذه المتغيرات بين بداية ونهاية ما يعرف بجزء توصيف المتغيرات المشتركة (DECLARE SECTION). كما يجب أن يتبع أية تعليمة تتعاطى مع قاعدة البيانات بالكلمتين المحجوزتين (EXEC SQL)، وذلك للتفريق بين التعليمات الخاصة بلغة البرمجة وبين التعليمات التى تتعاطى مع قاعدة البيانات. وبهذه الطريقة يتمكن مترجم لغة البرمجة من تجاهل تعليمات لغة الاستفسار البنائية فى أثناء ترجمة البرنامج (Program Compilation) وإرسالها بشكل مباشر لقاعدة البيانات فى أثناء تنفيذ البرنامج. وبعد الجزء المتعلق بتوصيف المتغيرات المشتركة، توجد تعليمة خاصة بلغة سى تهدف إلى قراءة المدخلات التى من المفترض أن يقوم المستفيد بإدخالها للبرنامج، وهى: رقم الرحلة (flight_no)، وتاريخها (day)، واسم العميل (customer_no). وبعد الحصول على بيانات العميل المزمع إجراء حجز له من قبل المستفيد (الذى يقوم بإجراء الحجز) ينفذ البرنامج تعليمة لغة استفسار تقوم بتحديث جدول الرحلات الجوية بحيث يزيد عدد المقاعد المباعة بمقعد واحد وذلك للرحلة الجوية المطلوب إجراء الحجز عليها. ويلاحظ هنا استخدام النقطتين المزدوجتين عند استخدام المتغيرات المشتركة ضمن تعليمات لغة الاستفسار البنائية، وذلك حتى يتمكن معالج لغة الاستفسار البنائية من التفريق بين مسميات المتغيرات المشتركة وبقية الكلمات والعبارات الواردة فى تعليمات لغة الاستفسار البنائية. بعد ذلك يقوم البرنامج بإضافة بيانات العميل والرحلة الجوية لجدول حجوزات العملاء. وينهى البرنامج المعاملة من خلال تنفيذ تعليمة التثبيت (Commit) وتعليمة الإطلاق (Release) التى تقوم بفصل المعاملة عن قاعدة البيانات وإطلاق الموارد التى تم حجزها فى قاعدة البيانات. وتجدر الإشارة إلى أن البرنامج السابق غير مكتمل؛ إذ يجب الاتصال بقاعدة البيانات التى سيقوم البرنامج بالتعامل معها وتزويدها برقم المستخدم وكلمة السر قبل التعامل الفعلى معها، ولهذا السبب قد تم وضع علامات تنقيط فى بداية البرنامج للدلالة على ذلك.

ويلاحظ فى المثال السابق أن تعليمة بدء المعاملة جاءت ضمنية، وذلك عند تنفيذ أول تعليمة من تعليمات لغة الاستفسار البنائية. أما تعليمة إنهاء المعاملة فقد جاءت ضمن تعليمة التثبيت (Commit). ويمكن أيضاً استخدام تعليمة الانسحاب فى المعاملات عوضاً عن تعليمة التثبيت كما يوضح المثال التالى الذى يتعامل مع قاعدة البيانات السابقة نفسها.

```
...
main{
...
    EXEC SQL BEGIN DECLARE SECTION;
            char flight_no[6], customer_name[20];
            char day; int temp1, temp2;
    EXEC SQL END DECLARE SECTION;
    scanf(flight_no, day, customer_name);
    EXEC SQL SELECT STSOLD,CAP INTO :temp1, :temp2
            FROM FLIGHT
            WHERE FNO = :flight_no AND DATE = :day;
    if temp1 = temp2 then {
             printf("no free seats");
             EXEC SQL ROLLBACK RELEASE;
             return(-1);}
    else {
             EXEC SQL UPDATE FLIGHT
                     SET STSOLD = STSOLD + 1
                     WHERE FNO = :flight_no AND DATE = :day;
             EXEC SQL INSERT INTO
                     FC(FNO, DATE, CNAME, SPECIAL)
                     VALUES(:flight_no, :day, :customer_name, null);
             EXEC SQL COMMIT RELEASE;
             printf("Reservation completed");
             return(0);}
}
```

يقوم البرنامج السـابق بنفس عمل برنامج الحجز الذى أسـلفنا شرحه أعلاه، إلا أنـه يقوم بإجراء عملية قراءة مسـبقة لكل من عدد المقاعـد المحجوزة وعدد المقاعد التى تمثل السـعة الكلية للرحلة الجوية التى من المفترض إجراء الحجز عليها، وذلك للتأكد من توافر مقاعد شـاغرة لم يتم حجزهـا قبل إجراء أى حجز جديد. وللتأكد من ذلك تتم قراءة عدد المقاعد المحجوزة وتخزينها فى المتغير (temp1) وتتم قراءة سعة الطائرة التى سـتقوم بالرحلـة الجوية وتخزينها فى المتغيـر (temp2). وبعد ذلك تتم مقارنة المتغيرين من حيث تساويهما. وعند تساوى قيمة المتغيرين فإن هذا يعنى عدم توافر مقاعد شاغرة لكون عدد المقاعد المباعة على الطائرة مساوياً لسعتها الكلية من الركاب مما يستدعى إنهاء المعاملة من خلال عملية الانسحاب (ROLLBAK). أما إذا لم يكن المتغيران متساويين، فإنه يمكن إجراء عملية الحجز وتثبيت النتيجة على قاعدة البيانات كما فى المثال السـابق. ويمكن تعديل البرنامج السـابق بحيث يقوم بإجراء حجوزات لأكثر من مقعد، وذلك من خلال تعريف متغير مشترك جديد من نوع الأعداد الصحيحـة، وليكن عدد المقاعد المطلوبة (no_of_seats) وتعديل البرنامج بحيث يتواكب مع هذا التعديل.

٩-١-١ التأكيد على خصائص المعاملات فى نظم إدارة قواعد البيانات:

عند تنفيذ المعاملات على قواعد البيانات فإنه لا بد أن تقوم نظم إدارة قواعد البيانات (Database Management Systems) بالمحافظة على خصائص المعاملات. ويتم ذلك من خلال بناء نظامين فرعيين ضمن أى نظام لإدارة قواعد البيانات، وهما: نظام التحكم فى التزامن (Concurrency Control Protocol) ونظام الاستعادة (أو التشافى) (Recovery Protocol). وعند تنفيذ المعاملات على قاعدة بيانات يقوم نظام إدارة قاعدة البيانات بتفكيك كل تعليمة (تتفاعل مع قاعدة البيانات) من تعليمات أية معاملة إلى تعليمات بسيطة تتكون من عمليات قراءة (Read) وعمليات كتابة (Write) بحيث إن كل عملية بسيطة تتفاعل مع عنصر واحد من عناصر قاعدة البيانات. وقد يكون العنصر حقلاً من حقول أحد سجلات جدول من جداول قاعدة البيانات، أو قد يكون سجلاً من سجلاتها، أو قد يكون كتلة (Block) من كتل جدول ما تمثل مجموعة من سجلاته، أو قد يكون حتى جدولاً كاملاً. ومع ذلك، فإن ماهية العنصر غير ذات أهمية من الناحية المنطقية عندما نتحدث عن نظام التحكم فى التزامن ونظام الاستعادة.

٩-١-١-١ نظام التحكم فى التزامن (Concurrency Control Protocol):

يتم بناء نظام التحكم فى التزامن لضمان خاصية العزلة (Isolation). ويُمكن ضمان هذه الخاصية من قبل نظم إدارة قواعد البيانات من خلال بناء نظام فرعى يقوم بجدولة العمليات البسيطة للمعاملات المختلفة، بحيث تبدو المعاملات التى تتبعها هذه العمليات وكأنها تنفذ بالتسلسل الواحدة تلو الأخرى على الرغم من تنفيذها بشكل متزامن، أى فى الوقت نفسه، على قاعدة البيانات. وبهذه الطريقة تنفذ المعاملات بشكل متزامن على قاعدة البيانات، ولكن تأثيرها على قاعدة البيانات يظهر كأن المعاملات تنفذ بشكل متسلسل الواحدة تلو الأخرى. وعلى افتراض أن قاعدة البيانات (التى تم البدء منها) صحيحة، تحتوى على بيانات تتوافق مع القيود المفروضة عليها، وأن كل معاملة قد تمت كتابتها بشكل صحيح يتوافق مع الضوابط (أو القيود) المفروضة على قاعدة البيانات فإن تنفيذ مجموعة من المعاملات على قاعدة البيانات بشكل متزامن سينتج عنه حالة جديدة صحيحة أيضاً لقاعدة البيانات طالما أن المعاملات قد تم تنفيذها بشكل يتماثل مع تنفيذها بشكل متسلسل ما، الواحدة تلو الأخرى دون أى تداخل (أو تزامن) بينها. وتُمكن نظرية التسلسل (Serializability Theory) من التحقق من أن أى طريقة لضبط التزامن تحقق خاصية العزلة من عدم تحقيقها لذلك. ومن

أشهر الطرق المتبعة فى بناء نظم التحكم فى التزامن التى تضمن تماثل تنفيذ المعاملات بشكل متزامن مع تنفيذها بشكل متسلسل طريقة الأقفال (أو الحجز المسبق) (Locking Protocols).

وباستخدام طريقة الأقفال يتم ربط كل عنصر من عناصر قاعدة البيانات بقفل، بحيث يمكن أن يوضع القفل فى وضع مخصص لقراءة العنصر، ويكون القفل فى هذه الحالة فى وضع حجز مشاركة أو قراءة (Shared or Read Lock)، أو فى وضع مخصص للكتابة على العنصر، ويكون القفل فى هذه الحالة فى وضع حجز استثناء أو كتابة (Exclusive or Write Lock). ونظراً لأن أى عمليتى قراءة تابعتين لمعاملتين مختلفتين لا تؤثران على محتويات قاعدة البيانات فإنه بالإمكان تنفيذهما بأى ترتيب كان. ولهذا السبب فإن قفل المشاركة يسمح بأن تقوم أكثر من معاملة بقراءة العنصر نفسه فى الوقت نفسه دون تضارب بين العمليتين. وفى هذه الحالة يقال إن عمليات قراءة العناصر تتوافق (Compatible) مع بعضها. وبناءً على ذلك فإنه يمكن تنفيذ عمليات القراءة على عناصر قاعدة البيانات بأى ترتيب كان، بمعنى أنه يمكن تنفيذها بشكل تبادلى (Commutable) على عناصر قاعدة البيانات دون التأثير على القيم المسترجعة من العناصر. أما عمليات الكتابة فإنها تؤثر فى محتويات قاعدة البيانات، ولذلك فإن ترتيب عمليات الكتابة على قيم العناصر من قبل المعاملات المختلفة مهم، إذ إن القيمة النهائية للعنصر أو القيمة المسترجعة منه تتوقف على القيمة التى قامت بكتابتها آخر معاملة على العنصر. لذا فإن ترتيب أى عمليتى كتابة تابعتين لمعاملتين مختلفتين منفذتين على عنصر ما يتضاربان، وذلك لكون ترتيبهما يؤثر فى القيمة النهائية للعنصر. وكذلك هو الحال عندما تكون إحدى العمليتين عملية قراءة، حيث إن القيمة المسترجعة من العنصر تتوقف على ترتيب عملية القراءة من حيث كونها قد تمت قبل عملية الكتابة أو بعداً منها. فعلى سبيل المثال، لو افترضنا وجود العنصر «س» (x) الذى يحتوى على القيمة «١٠» فى قاعدة البيانات ووجود معاملتين (t_1, t_2) فإن الترتيب التالى لعمليتى قراءة العنصر (Read) من قبل المعاملتين سينتج عنهما القيمة المسترجعة نفسها من العنصر، وهى «١٠» بغض النظر عن ترتيب العمليتين:

$$R_1[x]\ R_2[x] \equiv R_2[x]\ R_1[x]$$

أما إذا افترضنا أن العملية التابعة للمعاملة الأولى هى عملية كتابة (Write) على العنصر (x) بحيث تصبح قيمته «٥»، $W_1[x,5]$، فإن ترتيب العمليتين مهم لكون القيمة

المسترجعة من قبل عملية القراءة التابعة للمعاملة الثانية، $R_2[x]$، تختلف باختلاف ترتيبها فى التنفيذ. وستكون القيمة المسترجعة «٥» إذا نفذت عملية القراءة بعد عملية الكتابة من قبل المعاملة الأولى. أما إذا نفذت عملية القراءة قبل عملية الكتابة فستكون القيمة المسترجعة «١٠». وإذا افترضنا أن كلتا العمليتين التابعتين للمعاملتين هما عمليات كتابة فإن ترتيبهما مهم أيضاً؛ إذ إن القيمة النهائية للعنصر تعتمد على القيمة التى قامت بكتابتها آخر معاملة. فعلى سبيل المثال، إذا افترضنا أن عملية المعاملة الثانية هى أيضاً عملية كتابة على العنصر (x) ولكن بقيمة «٧»، $W_2[x,7]$، فإن قيمة العنصر (x) ستكون «٥» إذا كانت المعاملة الأولى هى آخر من قام بعملية الكتابة فى حين ستكون قيمة العنصر «٧» إذا كانت المعاملة الثانية هى آخر من قام بعملية الكتابة. ويعنى هذا أن أى عمليتين (Operations) تابعتين لمعاملتين مختلفتين على العنصر نفسه (x)، وليكونا ($O_1[x], O_2[x]$)، تتضاربان إذا وجد بينهما عملية كتابة. ويمثل مثل هذا التضارب كما يلى:

$$O_1[x]\ O_2[x] \neq O_2[x]\ O_1[x]$$

وباستخدام طريقة الأقفال فإنه يجب على كل عملية بسيطة تابعة لمعاملة ما أن تتحصل على قفل على العنصر الذى ستتفاعل معه (سواء من خلال قراءته أو بالكتابة عليه) وفى الوضع الذى يتناسب مع طبيعة العملية. وعندما لا تستطيع العملية أن تتحصل على القفل بالوضع المناسب نتيجة لوجود عملية أخرى تابعة لمعاملة أخرى قد سبق أن استحوذت على قفل على العنصر المطلوب وفى وضع يتضارب مع العملية المراد تنفيذها على العنصر، فإنه يتم تأخير تنفيذ العملية حتى انتهاء المعاملة التى تمتلك القفل على العنصر. ويعنى هذا أن المعاملة التى تتبعها التعليمة يجب أن تنتظر لحين فك القفل من قبل المعاملات الأخرى التى تتضارب معها (فى طبيعة القفل الذى وضع على العنصر) قبل أن تتمكن من مواصلة تنفيذ عمليتها. أما إن كانت العملية الواجب تنفيذها لا تتضارب مع العملية (أو العمليات) التى سبق أن استحوذت على قفل على العنصر (فى حالة وجود مثل هذا القفل على العنصر)، فإنه يتم وضع قفل على العنصر يبين أن المعاملة التى تتبعها العملية قد استحوذت على قفل على العنصر ومن ثم يتم تنفيذ العملية. وعندما ينتهى تنفيذ كافة العمليات التابعة لمعاملة ما (من خلال تنفيذ المعاملة لتعليمة التثبيت أو تعليمة الانسحاب)، يتم فك (أو تحرير) الأقفال كافة التى تم وضعها على العناصر التى تفاعلت معها المعاملة المنتهية. وبعد ذلك يتم السماح لأية معاملات أخرى متوقفة تنتظر انتهاء المعاملة من الحصول على الأقفال

التـــى تنتظر الحصـــول عليها ومن ثم يتم تنفيذ التعليمـــات التابعة لها. ويمكن تمثيل تنفيذ تعليمات أى معاملة، ولتكن (t_1)، بنظام الأقفال كما يلي:

$$rl_1[x]\ r_1[x]\ wl_1[y]\ w_1[y]\ rl_1[w]\ r_1[w]\ wl_1[z]\ w_1[z]\ c_1$$

ويعنى التمثيل الســـابق للمعاملة (t_1) أنه قد تم تفكيكها إلى عمليات بســـيطة تتكون مـــن عمليـــات قراءة وعمليات كتابة. كما يعنى التمثيل أن كل عملية بســـيطة لا بد أن يســـبقها عملية وضع قفل على العنصر الذى ســـتتفاعل معه العملية وفى الوضع الذى يتناسب مع طبيعة العملية قبل تنفيذ العملية على العنصر. فعلى سبيل المثال، يلاحظ وضـــع قفل قـــراءة ((Read Lock (rl)) علـــى العنصر (x) قبل تنفيـــذ تعليمة القراءة على العنصر. أما بالنسبة للعنصر (y) فيلاحظ أنه قد تم وضع قفل كتابة ((Write Lock (wl)) عليه قبـــل تنفيذ عملية الكتابة على العنصر (y)، مع تجاهـــل القيمة المفترض كتابتها علـــى العنصر فى هذا المثـــال. ويلاحظ أيضاً أن المعاملة قد تم إنهاء تنفيذها بتعليمة تثبيت ((Commit (c)).

وباســـتخدام نظام الأقفال فإن كل معاملة تمر فـــى مرحلتين من مراحل تنفيذها، وهمـــا: مرحلة النمـــو (Growing Phase) ومرحلة الاضمحلال (Shrinking Phase). ويتم فـــى مرحلة النمو وضـــع قفل على كل عنصر تحاول المعاملـــة التعامل معه، وذلك قبل التعامل الفعلى مع العنصر وفى وضع يتوافق مع طبيعة العملية. وبعد وضع القفل على العنصر يتم التفاعل معه (أى تنفيذ العملية المطلوبة عليه). أما فى مرحلة الاضمحلال فيتـــم فك الأقفال التى تم وضعها على العناصر التى قامت المعاملة بالتفاعل معها ولا يحـــق للمعاملة التفاعل مع (أو وضع أى أقفال إضافية على) أى عناصر جديدة. ويتم التعـــرف على مرحلة الاضمحلال من خلال تعليمـــة التثبيت أو تعليمة الإخفاق اللتين تمثلان انتهاء تعليمات المعاملات.

ويُعرف نظام الأقفال الســـابق بنظام «الأقفـــال ذى المرحلتين» (Two-Phase Locking (2PL)) لكون كل معاملة لا بد أن تمر بمرحلتين هما مرحلة النمو ومرحلة الاضمحلال، كما يضمن نظام الأقفال ذو المرحلتين تسلسل المعاملات، بحسب نظرية التسلسل، على الرغم من أن تنفيذ المعاملات يتم بشـــكل متزامن على قاعدة البيانات نفســـها. ويعد نظام الأقفال ذو المرحلتين الأكثر تطبيقاً فى نظم دارة قواعد البيانات؛ وذلك لسهولته وبساطته النسبية فى البناء (أو التنفيذ) ضمن نظم إدارة قواعد البيانات.

٩-١-١-٢ نظام الاستعادة (أو التشافى) (Recovery Protocol):

يتم بناء نظام الاستعادة (أو التشافي) لضمان خاصية النووية (Atomicity) وخاصية الـدوام (Durability). ويتم ضمان خاصية النووية من خلال تأكيد أن أى معاملة يجب تنفيذهـا باعتبارها وحدة منطقية متكاملة واحـدة، فإما أن يتم تنفيذ جميع عملياتها علـى العناصر التى تفاعلت معها المعاملة فى قاعدة البيانات وذلك عند انتهاء المعاملة ورغبتها فى تثبيت عملياتها، وإما أن لا يتم تنفيذ أى من عملياتها على أى من العناصر التـى تفاعلـت معها المعاملة فى حالة رغبة المعاملة فى الانسـحاب أو فى حالة وجود إخفـاق (أو عطل فى النظام). أما خاصية الدوام فيتـم ضمانها من خلال تأكيد أن أى معاملـة تم الانتهاء من تنفيذ جميـع عملياتها وتثبيت نتائجها على قاعدة البيانات سيستمر وجود نتائجها على قاعدة البيانات بغض النظر عن أى إخفاقات (أو أعطال) مستقبلية يتعرض لها النظام.

ومن أوسـع الطرق انتشاراً فى بناء نظم الاسـتعادة هى تلك المبنية على ما يعرف بسـجل الوقائع (Log or Journal). وسجل الوقائع عبارة عن ملف متسلسل (Sequential File or Append file) يستخدم من قبل نظام إدارة قاعدة البيانات. ويحتوى سجل الوقائع على بعض البيانات الأساسـية للمعاملات مثل أرقامها وبداياتها ونهاياتها، بالإضافة إلى جميع التعديلات التى يتم إجراؤها على قاعدة البيانات من قبل المعاملات المختلفة. وكلما وجب إجراء تعديل على قاعدة البيانات تتم إضافة سـجلات مناسـبة وبشـكل متسلسـل، الواحد تلو الآخر، على سجل الوقائع، بحيث تعكس طبيعة هذه التعديلات. ويعنى هذا أن سجل الواقع يحتوى على تسلسل تصاعدى للتعديلات التى تم إجراؤها على قاعدة البيانات. وتعد سـجلات الوقائع المبنية على ما يعرف بالتدوين المسـبق للوقائع (Write-Ahead Logging) هى أكثر سجلات الوقائع استخداماً من قبل نظم إدارة قواعد البيانات. وكما يدل مسـمى هذا النوع من سجلات الوقائع فإن أى تعديل على قاعدة البيانات يجب أن يدون فى سجل الوقائع قبل أن يتم إجراء أى تعديل فعلى على قاعدة البيانات. ويتم ذلك من خلال تدوين صورتين لأى عنصر قيد التعديل فى سجل الوقائع، وهما: الصورة السـابقة (لعمليـة التعديل) (Before Image) والصورة اللاحقة (لعمليـة التعديل) (After Image). فالصورة السـابقة لعنصر ما تمثل حالة (أو قيمة) العنصر قبل إجراء التعديل عليه من قبل المعاملة، أما الصورة اللاحقة فتمثل حالة (أو قيمة) العنصر بعد إجراء عملية التعديل عليه من قبل المعاملة.

وتســتخدم الصور السابقة لضمان خاصية النووية، وذلك عند انسحاب المعاملات من خلال تعليمات الإخفاق أو فى حالة حدوث أعطال للنظام. فعند انسحاب معاملةٍ ما يجب، حســب خاصية النووية، إزالة كل التعديلات التى أحدثتها المعاملة وانعكست فعلياً على قاعدة البيانات. ويتم ذلك من خلال اســتخدام الصور الســابقة للعناصر التــى تم التعديــل عليها من قبل المعاملة. وكذلك هو الحــال عند تعطل النظام حيث تســتخدم الصور الســابقة لضمان خاصية النووية، ولكن لإزالــة التعديلات التى قد أحدثتها المعاملات كافة التى تمت مقاطعة تنفيذها نتيجة للعطل الذى أصاب النظام. وتعــد المعامــلات التى تمت مقاطعتها غيــر منتهية التنفيذ لا مــن حيث تثبيت جميع نتائجها ولا من حيث انســحابها وإلغاء جميع نتائجها مــن قاعدة البيانات، مما يعنى عدم ســلامة النتائج التى عكستها هذه المعاملات على قاعدة البيانات نتيجة لمقاطعة تنفيذهــا الكامل حتى انتهائها. وتســتدعى مقاطعة تنفيذ المعامــلات إزالة أى تأثير للتعديــلات التى أحدثتها على قاعدة البيانات من خلال اســتخدام الصور الســابقة للعناصر التى تفاعلت معها هذه المعاملات.

أما الصور اللاحقة فتســتخدم لضمان خاصية الدوام التى تتطلب استمرار نتائج المعامــلات التى تم تنفيذها وانتهت بتثبيت نتائجهــا على قاعدة البيانات عند حدوث أعطــال للنظام. فعند حدوث أى عطل يتم الرجوع لســجل الوقوعات بهدف التعرف علــى المعامــلات المنتهية بعمليات تثبيــت. ولكل واحدة من هذه المعاملات تســتخدم الصور اللاحقة المدونة فى ســجل الوقوعات لتثبيت الصور اللاحقة للعناصر التى تم التعديل عليها من قبل المعاملة.

وباستخدام سجل التدوين المسبق للوقائع يتم إجراء استرجاع قاعدة البيانات لحالة ســليمة، بعد حدوث أية عطل للنظام، بشكل يضمن كلاً من خاصية النووية وخاصية الدوام من خلال المرور على سجل الوقوعات بثلاث مراحل، وهى كما يلى:

١- **مرحلة التحليل (Analysis Phase)**: يتم فى هذه المرحلة المرور على سجل الوقوعات من البداية وحتى النهاية، وذلــك للتعرف على جميع المعاملات المنفذة على قاعدة البيانــات ووضعها التنفيذى من حيث انتهائها من خلال عمليات تثبيت أو عمليات انسحاب أو بشكل غير طبيعى نتيجة للعطل.

٢- **مرحلــة إلغــاء التعديــلات (Undo Phase)**: يتم لكل معاملــة منتهية بتعليمة إخفاق أو بشــكل غير طبيعى (لا يوجد لها تعليمة تثبيت أو إخفاق فى ســجل الوقوعات) اســتخدام الصور الســابقة لإلغاء التعديــلات التى أحدثتها المعاملــة على قاعدة

البيانات. وتتم هذه المرحلة بشكل عكسى على سجل الوقوعات حيث يتم إلغاء تعديلات المعاملات ابتداءً من نهاية السجل، بشكل متسلسل، وانتهاءً ببدايته. ويعنى هذا أن عملية إلغاء التعديلات تتم بشكل يعاكس، من الناحية التاريخية (أو الزمنية)، التسلسل الزمنى للتعديلات التى تم إجراؤها على عناصر قاعدة البيانات من قبل المعاملات المنسحبة أو المنتهية بشكل غير طبيعى.

٣- **مرحلة استعادة التعديلات (Redo Phase):** يتم لكل معاملة منتهية بتعليمة تثبيت إعادة تثبيت التعديلات التى أجرتها المعاملة على قاعدة البيانات باستخدام الصور اللاحقة للعناصر. وتتم هذه المرحلة بشكل تصاعدى على سجل الوقوعات، وذلك من بدايته وحتى نهايته. ويعنى هذا أن عملية استعادة التعديلات تتم بشكل يتوافق من الناحية التاريخية (أو الزمنية) مع التسلسل الزمنى للتعديلات التى تم إجراؤها على عناصر قاعدة البيانات من قبل المعاملات التى تم تثبيت نتائجها.

ومع مرور الزمن قد يصبح سجل الوقوعات طويلاً للغاية، بحيث تستغرق عملية الاستعادة وقتاً طويلاً جداً بعد حدوث أعطال للنظام. ليس هذا فحسب وإنما قد يصبح سجل الوقوعات طويلاً جداً، بحيث يأخذ مساحة تخزينية كبيرة يتعذر معها تخزينه على القرص الصلب (أو الأقراص الصلبة) للحاسب الآلى. ولهذا السبب فإنه من الضرورى الحد من حجم سجل الوقوعات قدر ما أمكن ذلك. ومن الطرق المتبعة للحد من حجم سجل الوقوعات ما يعرف بنقاط الاختبار (Checkpoints). والفكرة الرئيسية وراء نقاط الاختبار هى أن يقوم نظام إدارة قاعدة البيانات بوضع علامة معينة فى سجل الوقوعات، تسمى نقطة اختبار (Checkpoint)، تدل على أن كافة المعاملات التى تقع قبل نقطة الاختبار قد تم الانتهاء منها بشكل كامل، سواء من خلال تثبيت نتائجها على قاعدة البيانات، إذا كانت منهية بعمليات تثبيت، أم من خلال إلغاء نتائجها من قاعدة البيانات، إذا كانت منتهية بتعليمات انسحاب أو كانت مخفقة لسبب ما. ونظراً لأن هذه المعاملات تعد منتهية بالكامل فإنه لا يجب على نظام إدارة قاعدة البيانات النظر فى آثارها على قاعدة البيانات عند حدوث عطل للنظام. أما بالنسبة للمعاملات التى لها سجلات، فى سجل الوقوعات، تقع بعد نقطة الاختبار فإنه يتوجب النظر فيها فى أثناء عملية الاستعادة. وبهذه الطريقة يمكن الحد من عدد المعاملات التى يجب العمل على استعادتها بعد حدوث أى عطل للنظام كما يمكن تقليص حجم سجل الوقوعات من خلال الاستغناء عن المساحة التخزينية المخصصة لسجلات الوقوعات التى تقع قبل نقاط الاختبار من خلال عملية تسمى عملية جمع

النفايــات (Garbage Collection). وتختلف نظــم إدارة قواعد البيانات فى الطرق التى تتبعها لوضع نقاط الاختبار والتعامل معها، ولكنها تتفق جميعاً على المفهوم الرئيســى لنقاط الاختبار.

٩-١-٢ الزنادات والإجراءات المتكررة (Triggers and Routines):

لم يكن من ضمن لغة الاستفســار البنائية قبــل مقياس (SQL-99) دعم للإجراءات (Procedures) أو الدوال (Functions) التى من الممكن أن يقوم المستفيدون بتعريفها، وذلك على الرغم من أن نظم قواعد البيانات المتوافرة على المســتوى التجارى توفر إمكانية تعريف مثل هذه الإجراءات والدوال. وما الزنادات (Triggers) إلا إجراءات (Routines) ذات مســميات، فإن كل زناد يتكون من مجموعة من تعليمات لغة الاستفسار البنائية. وتقع الزنادات تحت تحكم نظام إدارة قواعد البيانات بحيث يتم تنفيذ التعليمات التى يتكون منها الزناد عندما تتحقق الشروط التى يتطلبها تنفيذ الزناد. ويرتبط كل زناد بحدث معين يؤدى إلى تنفيذه، وهذه الأحداث هى تعليمات التعديل (الإضافة والحذف والتحديــث). ولأن هذه التعليمات تقوم بتغيير محتــوى قاعدة البيانات ومن ثم حالة قاعدة البيانات، فإن الشــروط المرتبطة بالزنادات قد تتحقق عند التعديل على قاعدة البيانات، ثم يقوم نظام إدارة قاعدة البيانات بتنفيذ الزنادات التى يتم تحقق شــروط تنفيذها.

وتعتبر الزنادات أحد مكامن القوة فى نظم إدارة قواعد البيانات، حيث يتم كتابتها مرة واحدة، ومن ثم يتم التحكم فى تنفيذها من خلال نظام إدارة قاعدة البيانات. وبهذه الطريقة يتم التحكم فى فرض تكامل البيانات وتناسقها بشكل أكبر، هذا بالإضافة إلى تقليص عدد التعليمات التى يجب كتابتها، بشــكل متكرر، من قبل التطبيقات المختلفة لتطبيق محتويات الزناد نفسها. ويتكون كل من الزنادات والإجراءات من مجموعة من التعليمــات الإجرائية إلا أن تنفيذ الزنادات يتم تلقائياً من قبل نظام إدارة قاعدة، فى حين لا يتم تنفيذ الإجراءات إلا بعد عملية مناداتها من قبل نظم التطبيقات.

٩-١-٢-١ الزنادات:

يتم تنفيذ الزنادات من قبل نظام إدارة قاعدة البيانات عندما تتحقق شروطها بغض النظــر عن التطبيق الذى أدى إلى تحقق شــروط تنفيذها. ومن الممكن أن يتسلســل تنفيذ الزنادات بحيث يؤدى تنفيذ أحد الزنادات إلى تنفيذ زناد آخر، وهكذا. وقد يتم

استخدام الزنادات، على سبيل المثال، للتحقق من قيود السلامة المرجعية أو لفرض بعض قواعد العمل المعمول بها فى المنظمة أو لتفعيل إجراء معين. وتتكون الزنادات من ثلاثة مكونات رئيسية، وهى:

- **الحدث (EVENT)**: ويمثل التغير فى حالة قاعدة البيانات الذى يؤدى إلى تفعيل (أو تنشيط) الزناد.
- **الشرط (CONDITION)**: الاختبار أو الاستفسار الذى يجب التحقق منه عند تفعيل الزناد. وعندما يكون الشرط استفساراً (Query) فإن الشرط يعد متحققاً (True) عندما تكون نتيجة الاستفسار غير خالية، أى يوجد ناتج للاستفسار.
- **الفعل (ACTION)**: مجموعة التعليمات التى سيتم تنفيذها عندما يتم تفعيل الزناد ويتحقق الشرط المصاحب له.

والشكل (٩-١) يوضح الشكل العام لتعريف الزنادات.

شكل (٩-١): الشكل العام لتعريف الزنادات

```
CREATE [OR REPLACE] TRIGGER trigger_name
    {BEFORE | AFTER} {INSERT | DELETE | UPDATE} ON Table_name
    [FOR EACH ROW [WHEN (Trigger_Condition)]]
    Trigger_body;
```

ومن الأمور التى يتم مراعاتها عند تعريف الزنادات هو التوقيت الذى سيقوم بتفعيل الزناد، بحيث إنه قد يتم تفعيل الزناد إما قبل أو بعد التعليمة المفترض أن تقوم بتفعيله، وقد تكون إما تعليمة إضافة أو حذف أو تحديث، وعما إذا كان سيتم تفعيل الزناد مرة واحدة على مستوى التعليمة (بشكل كامل) أو مرة واحدة على مستوى كل صف يتأثر بسبب تنفيذ التعليمة.

٩-١-٢-٢ الإجراءات المتكررة (Routines):

من الممكن أن تكون الإجراءات المتكررة فى (SQL) على هيئة إجراءات (Procedures) أو دوال (Functions). ويدخل لكل دالة قيمة (Parameter) واحدة وتعيد قيمة واحدة كذلك. أما الإجراء فيمكن أن يكون له قيم مدخلة، أو قيم مستعادة، أو كلا الاثنين معاً. والشكل (٩-٢) يوضح الشكل العام لتعريف الإجراءات المتكررة.

شكل (٩-٢): الشكل العام لتعريف الإجراءات المتكررة

```
{CREATE PROCEDURE | CREATE FUNCTION} routine_name
        ({parameter} [{, parameter} ...]})
        [RETURNS data_type result_cast]                    /* for functions only*/
        [LANGUAGE {ADA | C | COBOL | FORTRAN | MUMPS | PASCAL | PL1 | SQL}]
        [PARAMETER STYLE {SQL | GENERAL}]
        [SPECIFIC specific_name]
        [DETERMINISTIC | NOT DETERMINISTIC]
        [NO SQL | CONTAINS SQL | READS SQL DATA | MODIFIES SQL DATA]
        [RETURN NULL ON NULL INPUT | CALL ON NULL INPUT]
        [DYNAMIC RESULT SETS unsigned_integer]         /* for procedures only */
        [STATIC DISPATCH]                                  /* for functions only */
        routine_body;
```

٩-٢ قواعد البيانات الشيئية (Object-Oriented Database Systems):

تَســدُّ نماذج البيانات التقليديــة، وخاصة النموذج العلاقــى، الاحتياجات التقنية التــى يتطلبها تطوير نظــم التطبيقات المتعلقة بمكننة أعمــال المنظمات ذات الصبغة التقليديــة. إلا أن هذه النماذج تعانى بعض القصور عند محاولة تصميم وتنفيذ نظم تطبيقات أكثر تعقيداً مثل تطبيقات التصميم بمســاعدة الحاســب الآلى (Computer-Aided Design)، والتصنيع بمساعدة الحاسب الآلى (Computer-Aided Manufacturing)، والتجــارب العلمية، ونظم المعلومات الجغرافيــة (Geographical Information Systems)، وتطبيقات الوســائط المتعددة (Multimedia Applications)؛ على سبيل المثال لا الحصر. والســبب وراء هذا القصور يرجع إلى أن خصائص ومتطلبات هذه التطبيقات تختلف عــن خصائص ومتطلبات التطبيقات التقليدية مثل احتوائها على عناصر (أو أشــياء (Objects)) أكثــر تعقيداً، وتنفيذها لمعاملات أكثر طولاً من حيث الفترات الزمنية التى تحتاج إليها للتنفيذ، وحاجتها لأنواع بيانات جديدة لحفظ الصور والأصوات والبيانات النصية الطويلة بالإضافة إلى حاجتها لإجراء عمليات غير تقليدية على أنواع البيانات الجديدة.

وقد تم اقتراح النموذج الشيئى لتلبية احتياجات التطبيقات الأكثر تعقيداً، مثل تلك المذكورة أعلاه. ويوفر النموذج الشــيئى المرونة من حيــث عدم التقيد بنوعية بيانات معينــة أو تعليمات محددة للتعامل مع محتويات قاعــدة البيانات مقارنة بنظم قواعد البيانــات التقليدية. ومن المزايا الرئيســية للنموذج الشــيئى تمكين مصممى قاعدة

البيانات من توصيف هياكل الأشياء التى يرغبون فى نمذجتها بالإضافة إلى العمليات التـــى يمكن إجراؤها عليها. ومن الأمور المهمة الأخرى وراء اقتراح النموذج الشـــيئى فـــى نمذجة قواعد البيانات كون غالبية لغات البرمجـــة الحديثة تعتمد على المفاهيم الشيئية مما يشكل صعوبة فى تطوير نظم التطبيقات ما لم يكن لنظام قاعدة البيانات القدرة على التعامل مع هذه اللغات بشـــكل مباشـــر من خلال اســـتخدامها للمفاهيم نفسها. لذلك فإننا نجد أن قواعد البيانات الشيئية تتبنى العديد من المفاهيم التى تم تطويرها أساســـاً للغات البرمجة الشيئية. وتمثل المفاهيم الأساسية للنموذج الشيئى محور هذا الجزء من الكتاب.

٩-٢-١ مفاهيم الأشياء الموجهة (Object-Oriented Concepts):

إن أصل مصطلح الأشياء الموجهة (Object-Oriented) يعود إلى لغات البرمجة (Elmasri and Navathe, 2004). إلا أن المفاهيم الرئيســـية وراء هذا المصطلح نراها اليوم مطبقةً ليس فى لغات البرمجة فحســـب، وإنما فى نظم قواعد البيانات، وهندسة البرمجيات (Software Engineering)، وقواعـــد المعرفـــة (Knowledge Base)، ونظم الحاســـب الآلى بشكل عام. وكان من أول المفاهيم التى تم تطبيقها فى الأشياء الموجهة مفهوم الصنف (Class) الذى يقوم بتجميع الأشـــياء المتشابهة ضمن هيكل واحد يسمى الصنف. وبناءً علـــى ذلك ظهر مفهوم أنواع البيانات المجـــردة (Abstract Data Types) الذى يقصد منه إخفاء الهياكل الداخلية الخاصة بالأشياء التى تم تعريفها عن المستفيدين وفى الوقت نفسه توفير عمليات تمكنهم من التعامل مع هذه الأشياء. وأدى هذا إلى ظهور مفهوم التغليـــف (Encapsulation) الذى يقصد منه إخفاء المعلومات عن المســـتفيدين. ويمكن توضيح معنى هذا المفهـــوم إذا تصورنا أن الأعداد الصحيحة (Integers) (أو الحقيقية (Real)) عبارة عن أشياء لها هياكل، وأنه يمكن إجراء عمليات معينة عليها (مثل الجمع والطـــرح). فى هذه الحالة تكون الطريقـــة التى يتم تخزين هياكل هذه الأعداد عليها مخفاة عنا (كمســـتفيدين) ولا نعلم عنها شـــيئاً، ولكننا نستطيع تعريف الأعداد ضمن البرامـــج التى نقوم بكتابتها والتعامل مع هذه الأعداد من خلال العمليات التى يوفرها لنا نظام الحاسب الآلي. وهذا المثال شبيه بما يقصد به مفهوم التغليف إلا أن الأشياء التـــى يحاول أن يخفيها هذا المفهوم تكـــون عادة ذات هياكل بيانات أكثر تعقيداً. وقد تم لاحقاً ظهور مفاهيم إضافية للأشياء الموجهة من ضمنها تمرير الرسائل (Message Passing) والتوريث (Inheritance). وفيما يلى نستعرض أهم مفاهيم النموذج الشيئى.

٩-٢-١-١ مفهوم الشىء:

يتكون الشىء (Object)، سواء فى لغات البرمجة أو فى نظم قواعد البيانات الشيئية، من مكونين رئيسيين هما: حالة (أو قيمة) الشىء ((State (or Value))، وسلوكه (أو عمله) ((Behavior (or Operation)). وعلى الرغم من أن الأشياء قد تختفى بعد تنفيذ البرمجيات (فى لغات البرمجة) إلا أن الأشياء فى نظم قواعد البيانات الشيئية تتصف بالدوام (Durability) بحيث يبقى وجودها ضمن قاعدة البيانات حتى بعد انتهاء تنفيذ المعاملات التى تتعامل معها (ما لم تتم إزالتها بشكل صريح). ويعنى هذا أن قواعد البيانات الشيئية تقوم بتخزين الأشياء فى الذاكرة الثانوية بشكل دائم، وتسمح بأن تتم مشاركة التعامل مع هذه الأشياء من قبل البرامج والتطبيقات المختلفة. وتتطلب عملية المشاركة للأشياء ووجودها الدائم، وعلى خلاف الأشياء فى لغات البرمجة، استخدام التقنيات الخاصة بنظم قواعد البيانات مثل الفهرسة، ونظم التحكم فى التزامن، ونظم الاستعادة (أو التشافى). وحتى يمكن التمييز بين الأشياء المختلفة فى قاعدة البيانات فإن لكل شىء ذاتيته الخاصة به التى تميزه، وبشكل منفرد، عن بقية الأشياء فى قاعدة البيانات.

٩-٢-١-١-١ ذاتية الشىء (Object Identity):

لكل شىء يخزن فى قاعدة بيانات شيئية ذاتيته الخاصة به التى يقوم نظام إدارة قواعد البيانات الشيئية بإسنادها له. وتتمثل ذاتية الشىء بمعرف خاص به (Object Identifier) يتم توليده وإسناده إلى الشىء من قبل النظام. ولا تكون قيمة المعرف الخاصة بالشىء ظاهرة للمستخدمين، وإنما تكون خاصة بالنظام حتى يتمكن من التمييز بين الأشياء المختلفة المخزنة فى قاعدة البيانات، وحتى يتمكن من إدارة الارتباطات (أو العلاقات) بين الأشياء المختلفة. وأهم خاصية لذاتية الشىء هى عدم تغيرها بمرور الزمن. لذلك فإنه يجب فى نظم قواعد البيانات الشيئية أن تحتوى على آليات مناسبة تمكن من توليد ذاتية خاصة لكل شىء يخزن فى قاعدة البيانات. أما الخاصية الثانية لذاتية الشىء فهى عدم استخدامها أكثر من مرة بمعنى أنه عندما تتم إزالة شىء معين من قاعدة البيانات فإنه يجب عدم إعادة استخدام ذاتيته مع شىء آخر. والسبب وراء ذلك منطقى؛ إذ إن كل شىء فى الطبيعة له ذاتيته الخاصة التى لا يمكن إعادة استخدامها مرة أخرى للدلالة على أى شىء آخر.

وتعنـــى الخاصيتان أعلاه أن ذاتية الشـــىء يجـــب أن لا تعتمد على قيم خصائصه (Attributes)؛ لأن قيم خصائص الشـــىء قد تتغير بمرور الزمن، كما أن ذاتية الشـــىء يجـــب أن لا تعتمـــد على عنوان موقع (أو مكان) تخزين الشـــىء فـــى الذاكرة الثانوية للحاسب الآلى ، لأن عنوان الموقع قد يتغير مع مرور الزمن نتيجة لإعادة ترتيب الأشياء فى الذاكرة الثانوية للحاسب الآلى، ومن ثم إسنادها لأشياء أخرى فى قاعدة البيانات. ويعنى هذا أن قواعد البيانات الشيئية تقوم بتوليد وإسناد ذاتية فريدة لكل شىء يخزن فى قاعدة البيانات. وبمقارنة ذاتية الشـــىء والمفتاح الرئيسى للعلاقات (أو الجداول) فى النموذج العلاقى نجد أنه عندما تتغير قيمة المفتاح الرئيسى لسجل ما، فى النموذج العلاقى، فإن ذاتيته تتغير على الرغم من أنه ما زال يمثل الشـــىء نفســـه على أرض الواقع. بالإضافة إلى ذلك فإنه قد يكون للمفتاح الرئيســـى للشـــىء أكثر من مســـمى فى علاقات قاعدة البيانات مما يصعب معه الجزم بأن الشـــىء هو ذاته فى العلاقات المختلفة. فعلى سبيل المثال، قد يكون المفتاح الرئيسى فى علاقة ما هو «رقم الموظف» (Emp_No)، فـــى حين يكـــون فى علاقة أخرى هو «رقم الســـجل المدنى» (Emp_SSN)، اللذان يمثلان على أرض الواقع الخاصية نفســـها. لذلك فإن نظم البيانات الشـــيئية تتغلب على هاتين المعضلتين من خلال إســـنادها إلى ذاتية خاصة لكل شىء يعرف فى قاعدة البيانات وفق الخاصيتين المذكورتين أعلاه.

٩-٢-١-١-٢ حالة (أو قيمة) الشىء (State (or Value) of an Object):

حالة الشـــىء هـــى القيم الفعلية لخصائصـــه وللعلاقات التى تربطه مع الأشـــياء الأخرى فى قاعدة البيانات فى لحظة زمنية ما. ويعنى هذا أن حالة الشىء تتغير من وقـــت لآخر فى أثناء فترة حياته (أو تواجده). ويتم تحديد حالة الشـــىء فى وقت ما من خلال القيم التى تحتويها خصائصه بالإضافة لارتباطاته مع الأشـــياء الأخرى فى قاعدة البيانات. وتتغير حالة الشـــىء من وضع إلى آخر حســـب العمليات التى تنفذ عليه وتغير من سلوكه (أو عمله).

٩-٢-١-١-٣ سلوك (أو عمل) الشىء (Behavior (or Operation) of Objects):

سلوك الشىء يمثل تفاعله مع العالم الخارجى من خلال العمليات التى تنفذ عليه. ويعتمد ســـلوك الشىء على الحالة التى يوجد عليها وعلى طبيعة العملية المنفذة عليه. أما العملية فما هى إلا فعل يقوم به الشىء على حالته وإرجاع نتيجة الفعل للمستفيد (أو التطبيق أو الشىء) الذى قام باستدعاء العملية.

فعلى سبيل المثال، لنفترض وجود الطالب «أحمد صالح» ممثلاً كشىء ضمن قاعدة بيانات شيئية. فى هذه الحالة، يكون للشىء الذى يمثل الطالب «أحمد صالح» ذاتيته التى تميزه عن بقية الأشــياء فى قاعدة البيانــات، وفيهم الطلبة الآخرون، ومجموعة مــن الخصائص التى تميز الطالب مثل اســمه، وعنوانه، وتخصصه، وما إلى ذلك من خصائص أخرى تتعلق بالطلبة. وتتمثل حالة الشــىء «أحمد صالح» فى القيم الفعلية لخصائصــه فى لحظة مــا بالإضافة إلى العلاقات التى تربطه بالأشــياء الأخرى فى قاعدة البيانات. أما سلوك الشىء «أحمد صالح» فيتمثل من خلال تفاعله مع العمليات التى تجرى عليه مثل حســاب «المعدل التراكمي» أو حســاب «العمر». وبناءً على هذا، فإن الشىء «أحمد صالح» عبارة عن حزمة (Package) تتكون من حالة الشىء (أى قيم خصائصه وارتباطاته) وسلوكه (أى تفاعله تجاه العمليات التى تنفذ عليه).

٩-٢-١-٢ الفئة (أو الصنف) (Class):

عند حديثنا عن نموذج كينونة-علاقة تم التفريق بين الكينونة وفئة الكينونة، وذلك لإزالة أى التباس بين المقصود بحالة (أو واقعة) معينة من الحالات (أو الوقوعات) وبين فئــة الكينونة التى تتبعها الحالــة (أو الواقعة). وكذلك هو الحال فى قواعد البيانات الشــيئية: إذ يتم التفريق بين حالة (أو واقعة) من حالات (أو وقوعات) الشىء (Object Instance) وبــين الصنف (Object Class) الذى تتبعه الحالة (أو الواقعة). ويعنى هذا أن الحالة الواحدة من حالات الشىء لها وجودها فى وقت ما، وقد تحتل مساحة محددة فى الطبيعة، إلا أن الصنف عبارة عن شــىء مجــرد (Abstract) يعبر عن مجموعة من الأشياء التى تشترك فى مخططات هياكلها (Structure) وسلوكها (أو تفاعلها) (Behavior) مــع ما حولها. وقد يتســاءل المرء عن الفرق بــين الكينونة، كما تم تعريفها فى نموذج كينونة-علاقة، والشــيء، كما تم تعريفه فى النموذج الشــيئي. والواضح أنه بالإمكان تمثيل كل كينونة فى نموذج كينونة-علاقة على أنها شىء فى النموذج الشيئى. إلا أنه بالإضافة إلى تخزين المعلومات المتعلقة بحالة الكينونة (التى تتمثل فى قيم خصائصها والعلاقات التى تربطها مع الكينونات الأخرى)، فإن للشــىء سلوكاً يتمثل فى عمليات من الممكن أن تنفذ عليه بغية معرفة الحالة التى هو عليها أو للتغيير فيها.

ويمثل الشكل رقم (٩-٣) فئة (أو صنف) من الأشياء وهم الطلبة، وحالة (أو واقعة) مــن وقوعات هذه الفئة (أو الصنف) وهى الطالــب «أحمد صالح». ويظهر فى أعلى

الشـكل، الممثل للصنف، اسـم الصنف تتبعه قائمة بالخصائص المتعلقة به (كما تظهر فى الجزء الأوسـط من الشكل). أما فى أسـفل الشكل فتظهر قائمة بالعمليات التى من الممكن تنفيذها على هذه الفئة (أو الصنف) من الأشياء.

ويمثل صنف الطلبة، فى هذا المثال، مجموعة حالات الطلبة التى لها مخطط هيكل مشـترك وسلوك مشـترك أيضاً. فكل حالات الطلبة تشترك فى خصائص «الاسم» و«تاريخ الميلاد» و«العنوان» و«رقـم الهاتف» و«التخصص». كما أن كل حالات الطلبة لها السـلوك نفسه من خلال العمليات المشـتركة التى بالإمكان أن تنفذ عليها وهى: «حسـاب عمر الطالب» (Calculate_AGE( ))، و«حسـاب المعدل التراكمى» (Calculate_GPA( ))، و«تغييـر العنوان» (Change_Address(Address)). وبذلك فإن كل صنف عبارة عن نمـوذج (أو مخطط) للحالات التى يمثلها. وكل حالة تَعرف الصنف الذى تتبعه. فحالة الطالب «أحمد صالح» تَعرِف أنها تتبع لصنف «الطلبة». كما أن الحالات التى تتبع لصنف ما من الممكن أن تشـارك فى علاقات متشـابهة مع أشياء أخرى. فعلى سبيل المثال، كل الطلبة يقومون «بالتسجيل» فى مواد دراسية. لذلك فإنه من الممكن أن يرتبط صنف الطلبة بعلاقة تسـمى «تسـجيل فى» (Register_for) مع صنف آخر يسمى «المادة الدراسية» (Course).

شكل (٩-٣): فئة (أو صنف) الطلبة وحالة (أو واقعة) منها

	فئة (أو صنف) Object Class	حالة (أو واقعة) Object Instance
اسم الفئة (أو الصنف) Class Name	Student	Student
قائمة بخصائص الفئة List of Attributes	Name DOB Address Phone Major	Name = Ahmad Saleh DOB = 20/3/1975 Address = Malaz, Riyadh Phone = 474-2323 Major = Computer Science
قائمة بالعمليات List of Operations	Calculate_AGE() Calculate_GPA() Change_Address(Address)	

٩-٢-١-٢-١ أنواع العمليات:

يتم تحديد حالة الشــىء من خلال قيم خصائصه وارتباطاته مع الأشــياء الأخرى فى قاعدة البيانات. أما سلوك الشىء فيعتمد على حالته وعلى طبيعة العملية المنفذة عليه. والعملية عبارة عن اســتدعاء (Invocation) لفعل (Action) يقوم به شــىء ما فى النظام على شــىء آخر بغية الحصول على ردة فعل (Response) من الشــىء. ويمكن تصور عملية ما على أنها خدمة معينة يقوم بتوفيرها شىء ما لعملائه. وعند استدعاء العميل للعملية بغية الحصول على الخدمة فإنه يقوم بإرســال رسالة إلى الشىء مقدم الخدمــة، تبين طبيعة الخدمة المطلوبة. وبمجرد اســتلام طلــب الخدمة، يقوم مزود الخدمة بتنفيذ العملية وإعادة نتيجتها للعميل الذى قام بطلبها (أو استدعائها).

ويمكــن تصنيف العمليات التى بالإمكان اســتدعاؤها (أو تنفيذها) على الأشــياء المخزنة فى قواعد البيانات الشيئية إلى أربعة أصناف، وهى كما يلى:

١- **عمليــات إنشــاء (Constructor Operations)**: يقوم هذا النوع من العمليات بإنشــاء حالــة (أو واقعة) جديدة من حالات الفئة (أو الصنف). فعلى ســبيل المثال، يمكن إنشاء حالة جديدة من حالات الطلبة، كما يلى:

CREATE Student("Ahmad Saleh", 20/3/1975, "Malaz, Riyadh", "Computer Science")

ولتوافر هذا النوع من العمليات لجميع فئات (أو أصناف) الأشياء، فإنه لا يتم ذكره بشكل ظاهر (Explicit) عند تعريف فئات الأشياء.

٢- **عمليات استفســار (أو اســترجاع) (Query (or Selector) Operations)**: يقوم هذا النوع من العمليات باســترجاع حالة الشــىء دون إحداث أى تغييرات عليها. فعلى ســبيل المثال، يمكن استرجاع عنوان طالب معين باستخدام التعليمة (Get_Address( )). وهذه العملية غير موضحة فى الشكل رقم (٩-٣) لكون مثل هذه العملية ليس من الضرورى إدراجها ضمن عمليات الصنف؛ لأنها تســترجع قيمة خاصية أساسية من خصائص الصنف. أما عملية «حســاب عمر الطالــب» (Calculate_AGE( )) فهى من العمليات التى يجب إظهارها ضمن عمليات الصنف على الرغم من كونها عملية اســترجاع لا تؤثر فى حالة الطالب الذى تقوم العملية بحســاب عمره. والســبب وراء إظهار هذه العمليــة ضمن عمليــات الصنف يرجع إلى أن هذه العمليــة لا بد أن ترتبط بطريقة (Method) تبين بناء آلية حســاب عمر الطالب. وعادة تبنى هذه العملية على أســاس أنها اشــتقاق من خاصية تاريخ الميلاد بحيث يتم طرح تاريخ ميلاد الطالب من تاريخ

اليوم الذى تم فيه اســتدعاء العملية. وتاريخ اليــوم هو من القيم المتغيرة التى يمكن استرجاعها من نظام الحاسب الآلى وتسمى عادةً (SYS_DATE).

٣- **عمليات تعديل (Modification Operations)**: يقوم هذا النوع من العمليات بتغيير حالة الشىء. فعلى سبيل المثال، عندما يتم استدعاء العملية ((Change_Address(Address))، فإن هذه العملية ســتقوم بتغييــر القيمة المخزنة فى حقل عنــوان الطالب ليصبح مســاوياً للقيمة المدرجة ضمن محددات (Arguments) العملية (وهى القيمة المخزنة فى (Address)). ويتم إيضاح المحددات الظاهرة (Explicit) لأى عملية ضمن قوسين. أما فى حالة عدم وجود محددات ظاهرة فتترك العملية فى قوسين فارغين.

٤- **عمليات إتلاف (Destruction Operations)**: إن عمليات الإتلاف، وعلى النقيض من عمليات الإنشــاء، تقوم بإزالة حالة معينة من حالات الصنف من قاعدة البيانات. فعلى ســبيل المثــال، يمكن اســتخدام العمليــة (Destroy_Student( )) لإزالة حالات الطلبة. ولأن هذا النوع من العمليات متوافر فى جميع فئات (أو أصناف) الأشياء، وكما هو الحال فى عمليات الإنشــاء، فإنه لا يتم ذكره بشكل ظاهر (Explicit) عند تعريف فئات الأشياء.

كمــا يوجد هناك صنف خامس مــن العمليات، ولكن هذا الصنــف لا يطبق على (حــالات أو وقوعات) الأشــياء فردياً بل تطبق على الصنــف الذى تتبعه مجموعة من الأشياء تطبيقاً جماعياً. ويسمى هذا الصنف من العمليات «عمليات مساعدة» (Class Utilities) أو «عمليــات أصنــاف» (Class Operations). فعلى ســبيل المثال، تعد عملية «حساب عدد الطلاب فى تخصص» (Calulate_No_of_Students(Major)) من هذا النوع من العمليات؛ لأنه يجب تطبيقها على جميع الحالات المتوافرة من صنف الطلبة لمعرفة العدد الكلى للطلبة الدارسين فى تخصص معين.

٩-٢-١-٣ مفهوم التغليف (Encapsulation):

يعد مفهوم التغليف أحد المفاهيم المميزة للنظم واللغات الشيئية؛ إذ إن هذا المفهوم غير مطبــق فى نظم ونماذج قواعد البيانات التقليديــة. فمن المعتاد عليه فى النظم والنماذج التقليدية أن يكون هيكل مكونات قاعدة البيانات ظاهراً للمستفيدين وبرامج التطبيقات. فعلى ســبيل المثال، يتوافر فى نموذج البيانات العلاقى عمليات الاختيار والإضافــة والحذف والتحديث التــى يمكن تطبيقها على أى علاقــة (أو جدول) فى

قاعدة البيانات. ويعنى هذا أن هذه العمليات ذات صبغة عامة، غير مخصصة لشىء (Object) محدد من الأشـياء المخزنة فى قاعدة البيانات (العلاقية). كما أن العلاقات وحقولها تعد ظاهرة للمستفيدين وبرامج التطبيقات، بحيث يمكن التعامل معها بشكل مباشر من خلال التعليمات العامة التى يوفرها النموذج.

ويرتبط مفهوم التغليف بمفهوم «أنـواع البيانات التجريدية» (Abstract Data Types) ومفهـوم «إخفاء البيانـات» (Information Hiding) المعروفين فى لغات البرمجة الموجهة للأشـياء. ويقصد بهذه المفاهيـم، عند الحديث عن نظم قواعد البيانات الشـيئية، تعريف سـلوك فئة الشىء من خلال العمليات التى من الممكن أن تستدعى وتنفذ على الشـىء بحيث يكون هيكل (أو مكونات) الشـىء مخفياً عن العالم الخارجى وبشكل لا يسـمح بالتعامل معه إلا من خلال مجموعة من العمليات المحددة التى توفرها فئة (أو صنف) الشـىء. وبهذه الطريقة يكون المسـتفيدون على علـم بالواجهة التى يقدمها الشـىء للتعامل مع محتواه، وهى تتمثل فى اسم كل عملية يمكن تنفيذها على الشىء والمحـددات (أو المتغيـرات) (Parameters (or Arguments)) التى تتطلبها كل عملية. أما طريقة بناء الشىء والكيفية التى يقوم فيها بتنفيذ العمليات، بما فى ذلك هيكله (Data Structure) وبناء عملياته (Operation's Implementation)، فتكون مخفية عن المستفيدين. ويمكن تشـبيه العملية الواحدة بالإجـراء (Procedure) (أو الدالة (Function)) فى لغات البرمجة حيث يتم اسـتدعاء الإجراء (أو الدالة) من خلال مسماها بالإضافة إلى قيم محدداتهـا (أو متغيراتهـا). أما طريقة عمل الدالة أو هيـاكل البيانات التى تحتويها فتكون مخفاة عن المستفيدين من الإجراء (أو الدالة).

وتسمى واجهة العملية التى يتم تعريفها فى فئة الصنف «توقيعاً» (Signature). أما البناء الفعلـى لطريقة تنفيذ العملية داخل فئة الصنف فيسـمى «طريقة» (Method). ويتم اسـتدعاء إحدى الطرق المعرفة لشـىء ما من خلال إرسـال رسـالة (Message) للشـىء. وبعد ذلك يقوم الشـىء بتنفيذ الطريقة المخصصة لتنفيذ العملية المطلوبة وإعادة نتيجة التنفيذ للشـىء (أو المسـتفيد) الذى قام بإرسال الرسالة. وقد يتم فى أثناء تنفيذ طريقة ما إرسال رسالة من شىء إلى شىء آخر بشكل متداخل.

٩-٢-١-٤ التحميل الزائد (Polymorphism):

إن أصل «التحميل الزائد» وهى كلمة (Polymorphism) يعود إلى اللغة اليونانية، وهى كلمة (Polymorphic) التى تعنى نماذج متعددة، وتتكون من كلمتين هما: (Poly) ومعناها

تعدد، وكلمة (Morphos) ومعناه نموذج (أو شــكل). كما يسمى هذا المصطلح فى بعض الأحيــان بمصطلح «التحميل الزائد للعوامــل» (Operator Overloading). ويعنى مفهوم هذا المصطلح أن مســمى المعامل نفســه (أو الرمز) يمكن أن يرتبط بأكثر من طريقة تنفيذ للمعامل (أو الرمز)، وذلك حســب نوعية الأشــياء التى يطبق عليها المعامل (أو الرمز). ومن أكثر الأمثلة شيوعاً لشرح هذا المفهوم هو مثال عملية الضرب أو الجمع أو القسمة، فى لغات البرمجة. فعملية الجمع، على سبيل المثال، يرمز لها بالرمز «+». ويمكــن تطبيق معامــل الجمع على الأعداد الصحيحــة (Integers) والأعداد الحقيقية (Real) علــى حد ســواء، على الرغم مــن أن طريقة إجراء عمليــة الجمع فى الأعداد الصحيحــة تختلف عن طريقــة إجراء عملية الجمع فى الأعــداد الحقيقية من حيث الخوارزميات التى تستخدم لتنفيذ كل من العمليتين داخل الحاسب الآلى. ويتم تحديد طريقة التنفيذ، أى الخوارزمية المناســبة، لإجراء عملية الجمع من خلال نوعية بيانات العددين (أو المتغيرين) اللذين ستجرى عليهما عملية الجمع. ويعنى هذا أن رمز عملية الجمع فى لغات البرمجة قد يرتبط بأكثر من طريقة للتنفيذ، ومن ثم فهو محمل بشكل زائد (Overloaded). وبهذا الأســلوب نفسه يمكن تعريف العوامل (أو الرموز) فى نظم قواعد البيانات الشيئية؛ إذ إنه من الممكن أن يتم الدمج (أو المزج) بين شيئين من نوعية بيانات «صورة» (Image)، على ســبيل المثال، باستخدام رمز عملية الجمع؛ أو استخدام رمز عمليــة الجمع لإجراء عملية جمع بين أعداد مركبــة (Complex Numbers). وفى مثــل هاتين العمليتين يجب تعريف فئة صنف (Class) لكل من نوع الصور (Image) ونوع الأعداد المركبة (Complex Number) بحيث يتم تعريف عملية الجمع من ضمن العمليات التــى بإمكان كل صنف أن يقوم بتنفيذها. ويجب أيضــاً برمجة طريقة تنفيذ عملية الجمــع الخاصة بكل صنــف داخل فئة الصنف، بالإضافة إلــى تعريف نوعية بيانات الصور والأعداد المركبة وهياكلها داخل صنف كل منها.

٩-٢-١-٥ هرميات الأصناف والتوريث (Class Hierarchies and Inheritance):

توفر نظم قواعد البيانات الشــيئية، شأنها شــأن بقية أنواع نظم قواعد البيانات، القدرة على تصنيف الأشــياء حسب الفئة (أو الصنف) الذى تتبعه هذه الأشياء بحيث يتم تجميع الأشياء المتشابهة ضمن فئة (أو صنف) واحد، كما أسلفنا أعلاه. وتشترك الأشــياء التابعة لفئة (أو صنف) ما فى الخصائص نفســها وفى نفس أنواع العمليات التــى من الممكن أن تجرى عليها بالإضافة إلى نوعيــة الارتباطات التى من الممكن أن

ترتبــط بها مع أنواع أخرى فى قاعدة البيانات. وبالإضافة إلى ذلك فإن نظم قواعد البيانات الشيئية تسمح بتعريف أنواع (أو أصناف) جديدة تنحدر (أو تشتق) من أنواع (أو أصناف) سبق أن تم تعريفها. ويعرف هذا المبدأ فى نظم قواعد البيانات الشيئية بهرميات الأنواع (أو الأصناف).

وكما أسلفنا أعلاه أيضاً، يتم تعريف الفئة (أو الصنف) من خلال تسميتها وتعريف خصائصهــا والعمليــات (أو الطرق) التى من الممكن أن تجــرى (أو تنفذ) عليها. ويعد هــذا النوع من الفئات (أو الأصناف) أبســط أنواع الفئــات (أو الأصناف) التى يمكن توصيفهــا فى نظم قواعد البيانات الشــيئية. ومن أمثلة الفئــات (أو الأصناف) هذه فئــة الطالــب (Student) التى تم توصيفها فى الجــزء ٩-٢-١-٢. إلا أن النوع الفرعى ((Subtype (or Subclass)) يعد مفهوماً مفيداً عندما نرغب فى إنشاء فئة (أو صنف) جديد مماثــل لنــوع (أو صنف) موجود أصلاً، ولكنه غير مطابق لــه. فى هذه الحالة، يمكن تعريــف النوع الفرعى بحيث يرث الخصائص كافةً والعمليات المعرفة فى النوع الموجود (أو المعــرف) أصلاً، وإضافــة خصائص وعمليات جديدة للنــوع الفرعى غير موجودة فــى النوع المعرف أصلاً للنوع الفرعى. وبشــكل عام، يــرث النوع الفرعى الخصائص كافــةً والعمليات المعرفة فى النوع الرئيســى، بالإضافة إلــى إمكانية تعريف خصائص وعمليــات مرتبطة بالنوع الفرعى فقط. وبهذه الطريقة يجب تعريف وبناء الخصائص والعمليــات المتعلقة بالنوع الفرعى فقط دون الحاجة إلى تعريف وبناء العمليات المعرفة أصلاً بالنوع الرئيسي. فعلى سبيل المثال يمكن تعريف «صنف الطالب» (Student Class) و«صنف الموظف» (Employee Class). على أنهما صنفان فرعيان من «صنف الشــخص» (Person) لكل منهما الخصائص والعمليات التى يتفرد بها عن النوع الأصلى الذى استمد (أو اشتق) منه، كما هو موضح فى الشكل رقم (٩-٤).

وبالطريقة نفســها الموضحة فى الشكل رقم (٩-٤)، من الممكن أن يتم تعريف أنواع فرعية جديدة من الأنواع الفرعية التى تم تعريفها. وبهذه الطريقة تتكون لدينا هرمية مــن الأنواع (أو الأصناف) الفرعية، كل منها يرث الخصائص والعمليات المعرفة ضمن الأنواع (أو الأصناف) التى تعلوه فى الهرمية. ومن أهم ميزات هرميات الأصناف مبدأ إعادة الاســتخدام (Reusability) بحيث يمكن إعادة اســتخدام الخصائص والعمليات التى تعرف فى نوع (أو صنف) معين، فى جميع الأنواع (أو الأصناف) التى تستمد (أو تشتق) من النوع (أو الصنف) دون حاجة إلى تعريفها أو إعادة بنائها.

شكل رقم (٩-٤): مثال توضيحى للفئات (أو الأصناف) الفرعية

Person	اسم الفئة (أو الصنف) Class Name
Name DOB Address Phone Social_Identification_NO	قائمة بخصائص الفئة List of Attributes
Calculate_AGE() Change_Address (Address)	قائمة بالعمليات List of Operations

فئة فرعية (أو صنف فرعى) Object Subtype (or Subclass)		فئة فرعية (أو صنف فرعى) Object Subtype (or Subclass)
Employee	اسم الصنف الفرعى	Student
Salary Hire_Date	قائمة بالخصائص التى يتفرد فيها كل صنف فرعى	Major
Calculate_Bonus()	قائمة بالعمليات التى يتفرد فيها كل صنف فرعى	Calculate_GPA()

٩-٣ قواعد البيانات العلاقية-الشيئية (Object-Relational Database Systems):

يعد النموذج العلاقى، ونظم إدارته، من أكثر النماذج شيوعاً فى المنظمات الحديثة فـى وقتنا الراهـن فى بناء النظم المعلوماتية المبنية على نظـم قواعد البيانات، وذلك على اختلاف طبيعة الأعمال التى تزاولها هذه المنظمات. وتعزى أسباب نجاح النموذج العلاقى وسعة انتشاره لما يلى:

١- سهولة المفاهيم الأساسية للنموذج العلاقى واستناده إلى أسس رياضية صلبة (كما أوضح فى الفصل الرابع).

٢- يجسد النموذج العلاقى مفهوم عدم اعتمادية البيانات (كما أتضح فى الفصل الأول من الكتاب).

٣- يتوافر للنموذج العلاقى لغة تداول قوية وذات مواصفات قياسية (وهى لغة الاستفسار البنائية التى تم شرح مكوناتها الأساسية فى الفصل السابع والفصل الثامن).

٤- يتوافر للنموذج العلاقى نظم إدارة قواعد بيانات ذات تقنيات ناضجة تم تصميمها وتطبيقها لفترة طويلة من الزمن بما فى ذلك نظم للتحكم فى التزامن ونظم للتحكم فى الاستعادة (أو التشافى).

وعلى الرغم من نقاط القوة السابقة للنموذج العلاقي، إلا أن هذا النموذج لا يعد مناسباً للتعامل مع أنواع البيانات المعقدة مثل الصور، ولقطات الفيديو، والصوت، والبيانات الجغرافية؛ على سبيل المثال لا الحصر. لذلك تم ظهور نظم قواعد البيانات الشيئية بهدف التعامل مع مثل أنواع البيانات هذه. وعلى الرغم من أن نموذج البيانات الشيئى أخذ فى الانتشار وأن التقنيات المصاحبة له تبدو واعدةً، إلا أن هذا النموذج مازال يعانى نقصاً فى التقنيات المصاحبة له مقارنة بتلك المصاحبة للنموذج العلاقى. ومن ضمن هذه التقنيات: القدرة على التعامل مع البيانات بكفاءة عالية، ووجود نظم استعادة (أو تشافى)، ووجود نظم مرنة للتحكم فى التزامن.

ويمكن أن نستخلص أن لكل من النموذجين نقاط قوة تميزه على النموذج الآخر وأن نقاط قوة أحد هذين النموذجين تعد نقاط ضعف فى النموذج الآخر. وقد حدا هذا بالشركات المطورة لنظم إدارة قواعد البيانات إلى تبنى نظم مهجنة (Hybrid Systems) تهدف إلى الحصول على أفضل الميزات التى يوفرها كل من النموذجين. وأصبحت هذه النظم تسمى نظم قواعد البيانات العلاقية-الشيئية. كما أن غالبية الشركات المطورة لنظم إدارة قواعد البيانات العلاقية تتبنى ضمن النسخ الحديثة لمنتجاتها مفاهيم متوافقة مع مفاهيم نظم قواعد الشيئية مثل الشىء (Object)، والتغليف (Encapsulation)، والتحميل الزائد (Polymorphism)، والتوريث (Inheritance). ومن ضمن أكبر هذه الشركات التى تبنت هذا التوجه شركة أى بى إم (IBM) فى منتجها المعروف باسم دى بى تو (DB2)، وشركة إنفورميكس (Informix) فى منتجها المعروف باسم داينميك سيرفر (Dynamic Serever)، وشركة أوراكل (Oracle) فى منتجها أوراكل ٨ (Oracle 8) والنسخ اللاحقة لهذه النسخة من المنتج.

٩-٣-١ مفاهيم قواعد البيانات العلاقية-الشيئية:

يعد نظام إدارة قاعدة البيانات العلاقية-الشيئية نظاماً يدعم كلاً من خصائص النموذج العلاقى وخصائص النموذج الشيئى بشكل متكامل (Frank, 1995). وبذلك فإنه يمكن تعريف البيانات وفق النموذج العلاقى ووفق النموذج الشيئى والتعامل مع هذه البيانات من خلال واجهة مشتركة (مثل لغة الاستفسار البنائية (SQL)). أما البنية التحتية لهذه النظم فهى وفق النموذج العلاقى. ويعنى هذا أن قاعدة البيانات تبدو لمبرمجى نظم التطبيقات (وبقية المستفيدين من النظام) على أنها علاقية- شيئية، ولكنها فى الواقع تخزن ويتم التعامل معها داخلياً على أنها علاقية فقط. لذلك فإن هناك بعض التكلفة الإضافية فى هذه النظم نتيجة لضرورة المطابقة (أو التحويل) (Mapping) بين العلاقات (Relations) والأشياء (Objects). ويعزى السبب وراء هذه التكلفة الإضافية إلى أن هذه النظم قد تم بناؤها أساساً باعتبارها نظماً لإدارة قواعد البيانات العلاقية ولم تكن مصممة لدعم أى من الخصائص التى يوفرها النموذج الشيئى، ولكنه تم تطويرها لاحقاً لتدعم بعض خصائص النموذج الشيئي. لذلك فإننا نرى أن مثل هذه النظم تسمى أحياناً بالنظم العلاقية المطورة (Extended Relational)، وذلك للدلالة على هذا الواقع.

٩-٣-١-١ خصائص قواعد البيانات العلاقية-الشيئية:

توفر لغة الاستفسار البنائية ثلاثة أصناف من البيانات وهى: (١) الرقمية مثل الأعداد العشرية (Decimal) والأعداد الصحيحة (Integer)، (٢) والسلاسل الحرفية (Character)، (٣) والوقتية (Temporal) مثل التاريخ (Date) والوقت (Time). إلا أن الكثير من نظم التطبيقات الحديثة تتطلب تخزين وتداول أنواع أخرى من البيانات التى لا يمكن التعامل معها بسهولة من قبل نظم إدارة قواعد البيانات العلاقية مثل الوثائق، والصور، وبصمات الأصابع، والخرائط؛ على سبيل المثال لا الحصر. وللتعامل مع مثل أنواع البيانات هذه فإنه يتطلب من نظام إدارة قاعدة البيانات توفير إمكانية تعريف أنواع بيانات جديدة، حسب احتياجات تطبيقات المنظمة، بالإضافة إلى تلك المتوافرة فيها بشكل افتراضي. كما أن توفير إمكانية تعريف أنواع بيانات جديدة من قبل نظام إدارة قاعدة البيانات يتطلب بدوره من النظام توفير إمكانية تعريف عمليات جديدة يمكن إجراؤها على أنواع البيانات الجديدة، شبيهة بتلك العمليات التى يوفرها النظام بشكل افتراضى على أنواع البيانات الافتراضية، مثل الجمع والطرح بالنسبة للأعداد

والبيانات الوقتية، وعمليات المقارنة بالنسبة للسلاسل الحرفية. فعلى سبيل المثال، قد يستلزم تعريف نوع جديد من البيانات وليكن صورة (Image) (أو خريطة (Map)) توفير عملية جديدة يمكن تنفيذها على هذا النوع من البيانات مثل عملية التكبير (Zoom In) وعملية التصغير (Zoom Out).

كما أن إمكانية تعريف أنواع بيانات جديدة يتطلب من نظم إدارة قواعد البيانات توفير لغة فعالة للتعامل مع البيانات شبيهة بلغة الاستفسار البنائية بحيث تمكن من تعريف، وتداول البيانات. لأن الهدف من قواعد البيانات العلاقية-الشيئية هو دمج أفضل خصائص النموذج العلاقى وأفضل خصائص النموذج الشيئى ضمن نظام إدارة قاعدة البيانات، فإن أهم خصائص هذا الهجين الناتج من دمج النموذجين ما يلى:

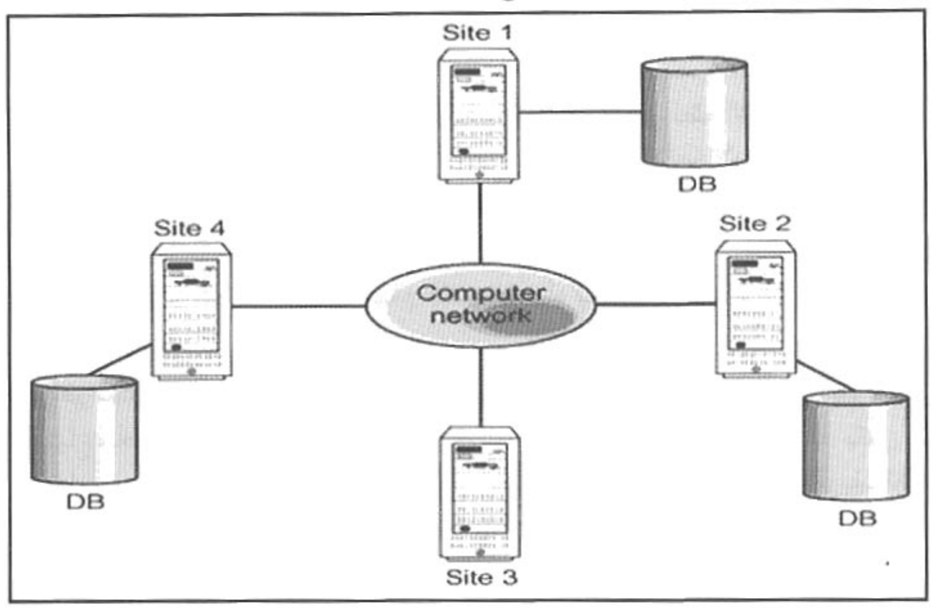
١- نسخة مطورة من لغة الاستفسار البنائية تمكن من تعريف وتداول البيانات، سواء كانت مخزنة فى جداول علاقية أو فئات أصناف (أو أشياء) (Object Types).

٢- توفير الدعم لخصائص النموذج الشيئى مثل التوريث، والتحميل الزائد، وتعريف أنواع بيانات جديدة (User Defined Data Types)، وطرق الوصول إلى الأشياء (Navigational Access).

٩-٤ قواعد البيانات الموزعة (Distributed Database Systems):

قاعدة البيانات الموزعة عبارة عن قاعدة بيانات منطقية واحدة موزعة مادياً على مجموعة من الحاسبات الآلية فى مواقع مختلفة ترتبط فيما بينها بشبكة اتصالات (Stallings, 1993). وتعد قواعد البيانات الموزعة مفيدة عندما يكون للمنظمة مجموعة من المواقع (أو الفروع) الموزعة فى مناطق مختلفة بحيث يتم توزيع البيانات بشكل أقرب ما يكون من المستفيدين الذين يتعاملون معها. ويمثل الشكل رقم (٩-٥) نموذجاً مبسطاً لقاعدة بيانات موزعة تتكون من أربعة مواقع (Sites) تحتوى ثلاثة منها على البيانات المكونة لقاعدة البيانات، فى حين لا يحتوى الموقع رقم ٣ (Site 3) على أى بيانات تتعلق بمحتويات قاعدة البيانات، وإنما يُستَخْدَم الموقعُ مصدراً للتعليمات التى تتفاعل مع قاعدة البيانات الموزعة على المواقع الثلاثة الأخرى. وترتبط المواقع الأربعة فيما بينها بشبكة اتصالات لنقل البيانات.

شكل رقم (٩-٥): نموذج مبسط لقاعدة بيانات موزعة



وتتــم إدارة قاعدة البيانات (الموزعة) بشــكل مركزى باعتبارهـا مورداً من موارد المنظمة، وفى الوقت نفسه تتم الاستفادة من المرونة التى تقدمها نظم قواعد البيانات الموزعــة فى تنفيــذ العمليات التى تجرى علــى قاعدة البيانــات. ويوجد العديد من الحالات التى تشــجع على اســتخدام نظم إدارة قواعد البيانات الموزعة التى يأتى من ضمنها ما يلى (Hoffer et al, 2002):

- من الطبيعى أن تنتشر الأقسام والإدارات فى المنظمات الحديثة فى مواقع متباعدة جغرافيــاً عن بعضها وفى بلدان مختلفــة أحياناً. وفى هذه الحالة تتولد رغبة عند هذه الأقســام والإدارات فى فرض صلاحياتها على نظمها المعلوماتية التى تستدعى توفــر هذه النظم المعلوماتية بمقربة من الأقســام والإدارات التى تتبعها هذه النظم حتى تتحكم فيها بشكل كامل.

- توجد هناك حاجة إلى المشاركة فى البيانات عندما تتوزع النظم المعلوماتيه والبيانات التابعة لهذه النظم فى أماكن مختلفة؛ حتى يتسنى اتخاذ القرارات التى تتعلق بأكثر من قسم وإدارة من الوحدات الإدارية التابعة للمنظمة. وتتجلى مثل هذه الحالة عند اندماج المنظمات (أو الشركات) بعضها مع بعض.

- إن وجود البيانات ونظم التطبيقات بالقرب من الأقسام والإدارات التى تحتاج إليها بشكل مكثف يكون فى الغالب أقل تكلفة من الناحية الاقتصادية؛ إذ إن إرسال كميات كبيرة من الأوامر الحاسوبية من مكان لآخر (عبر وسائل الاتصالات) يعد أمراً باهظ التكلفة. بالإضافة إلى ذلك فإن اعتماد نظم التطبيقات على نظم الاتصالات قد يكون مصدراً من مصادر المخاطرة نظراً لإمكانية تعطل نظم الاتصالات ولفترات طويلة. لذا فإن توزيع البيانات بحيث تكون فى مقربة من الوحدات الإدارية (والمستفيدين) الذين يحتاجون إليها بشكل مكثف يكون أكثر موثوقية مقارنة بتركيزها فى مكان واحد حتى لو تعطل جزء من نظام الاتصالات أو بعض الأجهزة التى تحتوى على بيانات المنظمة.
- يتوافر لدى المنظمات الحديثة أنواع مختلفة من نظم التطبيقات التى قد تعتمد على أنواع مختلفة من نظم إدارة قواعد البيانات، بحيث إن كل نوع منها قد يكون هو أفضل ما يكون للنظام الذى تم اقتناؤه من أجله. وفى هذه الحالة يمكن تعريف نظام قواعد بيانات موزع يعمل على ربط التطبيقات المختلفة مع بعضها.
- تعد تكرارية البيانات (Data Replication) إحدى الإستراتيجيات المتبعة لاستعادة البيانات التى تتعرض للتلف ولتمكين المستفيدين من مزاولة تعاطيهم مع البيانات حتى لو تعطل المصدر الرئيسى الذى تم تخزين البيانات عليه. وتعد تكرارية البيانات على أكثر من جهاز حاسب آلى أحد أشكال نظم قواعد البيانات الموزعة.

ويعد من الأهداف الرئيسية لنظم قواعد البيانات الموزعة توفير خدمة وصول المستفيدين للبيانات المخزنة فى مواقع مختلفة. وللوصول إلى الهدف يجب فى نظام إدارة قواعد البيانات الموزعة توفير ما يعرف بالشفافية المكانية (Location Transparency) التى تعنى أنه ليس من الضرورى أن يكون المستخدم على دراية بالمواقع التى تحتوى على البيانات التى يتعاطى معها من خلال التعليمات التى يصدرها للنظام، بل إن كل تعليمة يصدرها المستخدم سيتم إرسالها من قبل نظام إدارة قاعدة البيانات بشكل تلقائى (دون علم من المستخدم) للمواقع ذات الصلة بتنفيذ التعليمة. وفى الحالة المثلى لنظم قواعد البيانات الموزعة يكون المستخدم غير مدرك لطريقة توزيع البيانات ومواقعها، وأن كل البيانات على اختلاف مواقعها تشكل قاعدة بيانات منطقية واحدة. وفى مثل هذه الحالة المثلى فإن التعليمة الواحدة قد تقوم بتجميع بيانات موجودة مادياً فى مواقع مختلفة للحصول على النتيجة النهائية للتعليمة.

ومن الأهداف الرئيسية الأخرى لنظم قواعد البيانات الموزعة الاستقلال الذاتى (Local Autonomy) للمواقع المختلفة لقاعدة البيانات التى يقصد بها القدرة على إدارة كل موقع من مواقع قاعدة البيانات بشكل مستقل، بحيث يستطيع كل موقع العمل باستقلالية عن بقية المواقع، وذلك عند حدوث عطل لشبكة الاتصالات أو للمواقع الأخرى. وعند وجود الاستقلال الذاتى فإنه بإمكان كل موقع أن يتحكم فى البيانات المخزنة فيه، وأن يدير أمنها، وأن يقوم بتسجيل المعاملات التى ترد إليه وتنفيذها، وأن يقوم باستعادة بياناته عن حدوث عطل فيه دون وجود موقع مركزى يشرف على أعماله هذه. وبذلك فإن البيانات تعد مملوكة محلياً ضمن الموقع ومدارة من قبله، على الرغم من أن البيانات المخزنة فيه يمكن التعاطى معها من قبل مواقع أخرى.

ويوجد لقواعد البيانات الموزعة العديد من الميزات مقارنة بنظم قواعد البيانات المركزية. ومن بين أهم هذه الميزات ما يلى:

- **الموثوقية (Reliability) والتواجد (Availability):** تعرف الموثوقية، بشكل عام، على أنها احتمالية وجود النظام بشكل عامل فى لحظة زمنية ما؛ فى حين يعرف التواجد على أنه احتمالية وجود النظام بشكل عامل وبشكل متواصل ضمن مدى زمنى ما. وعلى النقيض من نظم قواعد البيانات المركزية التى يتأثر فيها جميع المستفيدين والتطبيقات عند تعطل النظام بحيث يصبح من المتعذر الوصول إلى كل البيانات المخزنة فى قاعدة البيانات، فإن حدوث عطل لموقع (أو أكثر) فى نظم قواعد البيانات الموزعة أو لجزء من شبكة الاتصالات لن يؤثر فى جميع المستفيدين والتطبيقات؛ وذلك لأنه من الممكن أن يتم التعاطى من الجزء الذى مازال عاملاً من قاعدة البيانات، ولكن بشكل وظيفى أقل مما كانت عليه قبل حدوث العطل. ويعنى هذا أن نظم قواعد البيانات الموزعة تقوم بتحسين كل من موثوقية وتواجد النظام.

- **التحكم الذاتى (Local Control):** يشجع توزيع البيانات المجموعات المختلفة من المستفيدين على ممارسة تحكمهم فى البيانات الخاصة بهم بشكل أكبر؛ مما يعنى تحسين تكامل البيانات وإدارتها. كما يمكن استخدام العتاد المادى المناسب من الحاسبات الآلية وملحقاتها مع البيانات الخاصة بكل مجموعة من المستفيدين وبما يتناسب مع احتياجاتهم من القدرات الحاسوبية. وبهذه الطريقة يمكن لكل مجموعة التعامل مع البيانات الخاصة بها وفق احتياجاتها الحاسوبية التى تتناسب معها، وفى الوقت نفسه، التعاطى مع البيانات التابعة للمجموعات الأخرى من المستفيدين عند الحاجة لذلك.

- **تخفيض تكلفة الاتصالات:** تُمكن نظم قواعد البيانات الموزعة من تخزين البيانات بشكل أقرب ما يكون من المستفيدين الذين يحتاجون إلى التعامل معها بشكل مكثف. ويعنى هذا تخفيض تكلفة استخدام وسائل الاتصالات مقارنة بنظم قواعد المركزية.

- **تحسين أداء النظام:** يمكن توزيع البيانات بشكل يلبى احتياجات المستفيدين فى كل موقع. وعند توزيع البيانات بهذا الشكل يصبح بالإمكان تنفيذ تعلميات المستفيدين فى مواقعهم دون الحاجة إلى استخدام وسائل الاتصالات أو تركيز تنفيذ التعليمات فى حاسب آلى واحد مما يحسن أداء النظام بشكل كبير. كذلك يمكن تجزئة التعليمات المعقدة وتنفيذ التعليمات الجزئية الناتجة على أكثر من حاسب آلى بشكل متزامن، أى فى الوقت نفسه، على أكثر من حاسب آلى مما يسهم فى تحسين أداء النظام بشكل أكبر.

- **سهولة التوسع فى النظام:** تتوافق نظم قواعد البيانات الموزعة مع حاجة المنظمات الحديثة للتوسع فى حجم بياناتها وتطبيقاتها الحاسوبية؛ إذ إنه بالإمكان زيادة عدد الحاسبات الآلية (لزيادة طاقتها التنفيذية) وطاقاتها التخزينية كلما استدعت الحاجة إلى التوسع. وهذه الطريقة تعد أكثر اقتصاداً من التوسع فى نظم قواعد البيانات المركزية التى تستدعى تغيير الحاسب الآلى (وبعض ملحقاته أحياناً). بالإضافة إلى ذلك فإن التوسع فى نظم قواعد البيانات الموزعة لا يؤدى إلى انقطاع الخدمة عن المستفيدين أو تأثر عملهم مقارنة بنظم قواعد البيانات المركزية التى قد تستدعى مثل هذا الانقطاع أو التأثر.

٩-٤-١ خيارات توزيع البيانات:

عند استخدام نظم قواعد البيانات الموزعة يجب تحديد الأماكن (أو المواقع) التى سيتم فيها تخزين كل مجموعة من البيانات المكونة قاعدة البيانات. ولأن الجدول أصغر وحدة منطقية يمكن الحديث عنها فى نظم قواعد البيانات، فإن الجدول يعد أصغر وحدة منطقية يمكن التحدث عنها عند توزيع البيانات على مواقع نظم قواعد البيانات الموزعة. كما يمكن تقسيم كل جدول إلى مجموعة من الجداول الجزئية بحيث يحتوى كل جدول جزئى على مجموعة من السجلات التى تتحلى بخصائص مشتركة فيما بينها. وفى هذه الحالة يعد كل جدول جزئى جدولاً مستقلاً بذاته عن بقية الجداول الجزئية المشتقة من الجدول الرئيسى نفسه. ويمكن توزيع جداول

قاعدة البيانات فـــى نظم قواعد البيانات الموزعة على المواقع المكونة للنظام وفق أربع إستراتيجيات هى:

- تكرار البيانات (Data Replication).
- التقسيم الأفقى (Horizontal Partitioning).
- التقسيم الرأسى (Vertical Partitioning).
- الجمع بين الخيارات السابقة.

٩-٤-١-١ تكرار البيانات:

يعد تكرار البيانات واحداً من الخيارات التى يزداد انتشارها يوماً بعد آخر. ويقصد بتكرار البيانات تخزين نســـخ كاملة من قاعدة البيانات فى أكثر من موقع. ويوفر هذا الخيار فرصة للانتقال مـــن قواعد البيانات المركزية إلى قواعد البيانات الموزعة ذات التكلفة الأقل، بحيث تكون البيانات قريبة مكانياً من المســـتفيدين منها. وتحسن هذه الإســـتراتيجية من إستراتيجيات توزيع البيانات من الاعتمادية والتواجد: إذ إن النظام سيســـتمر فى العمل مادام قد وُجِدَ موقع واحد (على الأقل) قيد العمل. كما يحســـن هذا الخيار أداء النظام لأن نتائج تعليمات الاستفســـار التى تصدر للنظام تكون قريبة من مكان صدور التعليمات نفســـها. إلا أن خيار تكرارية البيانات يعانى مشـــكلةً مع تعليمات التعديل إذ إن التعليمات يجب أن تنفذ فى جميع المواقع التى تحتوى على نسخ مـــن قاعدة البيانـــات، وذلك حتى تكون البيانات المخزنة فـــى المواقع المختلفة متوافقاً بعضها مع بعض. ولهذا الســـبب فإن تكرار البيانات يســـتخدم عادة فى نظم قواعد البيانات الموزعة التى يغلب فيها عمليات الاستفسار عوضاً عن تعليمات التعديل.

وبينما يمثل التكرار الكامل للبيانات إحدى النهايتين القصويين لإستراتيجية تكرار البيانـــات، فإن عدم التكرار لأى مـــن البيانات يمثل الجانب الآخر من النهايتين. وبين هاتين النهايتين القصويين توجد مســـاحة كبيرة للتكرار الجزئى للبيانات (Partial Data Replication) الـــذى يتم من خلاله تكرار بعض أجـــزاء قاعدة البيانات، فى حين لا يتم تكرار أجزاء أخرى. ويتم توزيع الأجزاء الرئيسية لقاعدة البيانات أو المستنسخة منها على مواقع محددة فى النظام. ويتم اختيار المواقع ودرجة التكرار بناء على الاعتمادية والتواجد المســـتهدفتين، بالإضافة إلى نوعيـــة وتكرارية العمليات المتوقع تنفيذها على النظام. وتعـــد عملية إيجاد التوزيع الأمثل للبيانات علـــى المواقع المختلفة للنظام من

العمليات المعقدة نسبياً، وذلك بسبب وجود العديد من العوامل التى يجب أخذها بعين الاعتبار عند إجراء عملية التوزيع.

٩-٤-١-٢ التقسيم الأفقى:

يقصد بالتقسيم الأفقى تجزئة الجدول إلى مجموعات من السجلات، بحيث تتحلى كل مجموعة من السجلات بخاصية (أو قيمة) محددة لحقل أو أكثر من حقول الجدول. وغالباً ما يتم التقسيم الأفقى للجداول وفق قيمة واحدة من حقول الجدول. فعلى سبيل المثال، من الممكن أن يتم تقسيم جدول أعضاء هيئة التدريس فى الجامعة الأهلية أفقياً وفق قيم حقل «رمز القسم» . وبهذه الطريقة يتم تقسيم الجدول إلى مجموعات من السجلات بحيث تتحلى كل مجموعة بقيمة محددة لقيمة حقل «رمز القسم». وبعد تقسيم الجدول إلى مجموعات يُمكن توزيع الأجزاء الناتجة بحيث تكون على أجهزة الحاسب الآلى التى تكون أقرب ما يكون من القسم العلمى الذى تتبعه كل مجموعة. وبهذه الطريقة يمكن أن تتعامل كل مجموعة من المستفيدين التابعين لقسم علمى ما مع بيانات أعضاء هيئة التدريس التابعين للقسم نفسه بشكل أسرع مما لو كانت البيانات مركزة فى موقع واحد.

٩-٤-١-٣ التقسيم الرأسى:

يستخدم التقسيم الرأسى لتوزيع الحقول التى يحتويها جدول ما على أكثر من جدول مع تكرار المفتاح الرئيسى للجدول الأساسى فى جميع الجداول الناتجة من عملية التقسيم. ومن أمثلة التقسيم الرأسى، وكما أسلفنا فى الجزء المتعلق بالتصميم المادى، تقسيم جدول الموظفين إلى جدولين: جدول يحتوى على المفتاح الرئيسى لجدول الموظفين، بالإضافة إلى الحقول التى تحتوى على البيانات المتعلقة بالجوانب الإدارية للموظفين، فى حين أن الجدول الثانى يحتوى على المفتاح الرئيسى لجدول الموظفين بالإضافة إلى الحقول التى تحتوى على البيانات المتعلقة بالجوانب المالية للموظفين. وبعد عملية التقسيم تتم عملية توزيع الجداول الناتجة على المواقع المناسبة فى النظام. وبهذه الطريقة يمكن الاستجابة للتعليمات المتعلقة بكل نوع من البيانات بشكل أسرع من الرد عليها عند تخزين جميع البيانات المتعلقة بالموظفين ضمن الجدول نفسه. ولأنه سيتم وضع بيانات كل إدارة على الجهاز التابع (أو الأجهزة التابعة) للإدارة؛ فإن هذا التوزيع سيوفر لكل إدارة استقلالاً ذاتياً يمكنها من التحكم والسيطرة على

البيانات التابعة لها مما يحسن من تناسق البيانات وتكاملها. بالإضافة إلى أن مثل هذا التوزيع سيزيد من المحافظة على أمن البيانات؛ إذ يُمكن من منح الصلاحيات للمستفيدين المختلفين على كل جدول من الجداول الناتجة من عملية التقسيم عوضاً عن منح الصلاحيات لهم على الجدول الأصلى كاملاً.

٩-٤-١-٤ الجمع بين خيارات التوزيع:

النوع الثالث للتقسيم يتضمن كلاً من تكرار البيانات والتقسيم الأفقى والتقسيم الرأسى أو أى توليفات أخرى للخيارات السابقة من توزيع البيانات. فعلى سبيل المثال، من الممكن تقسيم جدول الموظفين التابع لمنظمة ما أفقياً حسب الفروع التى يتبع لها الموظفون، وبحيث يتم توزيع الجداول الناتجة من عملية التقسيم على الحاسبات الآلية التابعة للفروع المختلفة كل حسب الموظفين التابعين له. وبعد ذلك يمكن تقسيم كل جدول، فى كل فرع، رأسياً حسب بيانات الموظفين الإدارية والمالية، على افتراض وجود إدارة للشئون الإدارية وإدارة للشئون المالية فى كل فرع. وبهذه الطريقة يمكن الاستفادة من الميزات التى توفرها قواعد البيانات الموزعة التى يأتى من أهمها سرعة الوصول إلى البيانات والتعامل معها من قبل كل إدارة، هذا بالإضافة إلى الاستفادة من الميزات الأخرى مثل المحافظة على الاستقلال الذاتى وأمن البيانات. وتجدر الإشارة إلى أن الجمع بين خيارات توزيع البيانات، وخاصة التقسيم الرأسى والتقسيم الأفقى للبيانات، لا يعد مقصوراً على قواعد البيانات الموزعة، ولكنه يستخدم أيضاً فى قواعد البيانات المركزية أيضاً. ويستخدم مثل هذا الجمع بين الخيارات، فى قواعد البيانات المركزية، من قبل إداريى قواعد البيانات فى الكثير من الأحيان لتحسين أداء نظم إدارة قواعد البيانات (Agarwal et al, 2004).

المراجع

1- Agrawal, S., V. Narasayya and B. Yang. "Integrating Vertical and Horizontal Partitioning into Automated Physical Database Design," *Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data*, Paris, France, June, 2004.

2- Bernstein, P. A., V. Hadzilacos and N. Goodman. "Concurrency Control and Recovery in Database Systems," Addison-Wesley, 1987.

3- Bruce, T. A. "Designing Quality Databases with IDEFIX Information Models," New York: Dorset House, 1992.

4- Cannan, Stephen and Gerard Otten. "SQL - The Standard Handbook," McGraw-Hill Book Company, 1993.

5- Codd, E. F. "A Relational Model of Data for Large Relational Databases," Communications of the ACM, Vol. 13, June 1970, pp. 377-397.

6- Connolly, Thomas and Carolyn Begg. "Database Systems: A Practical Approach to Design, Implementation, and Management," 3rd Edition, Essex, England: Pearson Education Limited, 2002.

7- Efraim, Turban, Ephraim Mclean and James Wetherbe. "Information Technology for Management," 3rd Edition, New York: John Wiley & Sons, Inc, 2002.

8- Elmasri, Ramez and Shamkant Navathe. "Fundamentals of Database Systems," 4th Edition, Pearson Education, Inc. 2004.

9- Fleming, C. and B. von Halle. "Handbook of Relational Database Design," Reading, MA: Addison-Wesley, 1989.

10- Frank, M. "Object-Relational Hybrids," DBMS, July 1995, pp. 46-56.

11- Garcia-Molina, Hector, Jeffery Ullman and Jennifer Widom. "Database Systems: The Complete Book," New Jersey: Prentice Hall, Inc. 2002.

12- Gray, Jim and Andreas Reuter. "Transaction Processing: Concepts and Techniques," Morgan Kaufmann, San Mateo, CA,1992.

13- Hoffer, Jeffrey, Mary B. Prescott and Fred R. McFadden. "Modern Database Management," 6th Edition, Upper Saddle, NJ: Prentice Hall, Inc. 2002.

14- Inmon W. H. "What Price Normalization," Computer World, Vol. 27, No. 31, 1988.

15- Martin, James. "Information Engineering: Book I, Introduction," New Jersey: Prentice-Hall, Inc, 1989.

16- Pargue, Cary and Michael Irwin. "Access 2002 Bible," 2001.

17- Ramakrishnan, Raghu and Johannes Gehrke. "Database Management Systems," 3rd edition, McGraw-Hill Higher Education, 2003.

18- Rogers, U. "Denormalization: Why, What and How?" Database Programming and Design 2, pp. 46-53, December 1989

19- Stallings, William. "Local and Metropolitan Area Networks," 4th Edition, Macmillan Publishing Company, 1993.



الملاحق

ملحق رقم (١): حالة دراسية - قاعدة بيانات الجامعة الأهلية

ملحق رقم (١)-١ قواعد العمل المعمول بها فى الجامعة

تنفـذ الجامعة الأهلية مجموعة من المواد الدراسـية فى كل فصل دراسـى. ومن قواعد العمل المتبعة فى الجامعة الأهلية ما يلى:

١- يوجد فى الجامعة عدد من الأقسام الدراسية، ولكل قسم (Department) من الأقسام العلمية رمز (Department_ID) يميزه عن بقية الأقسام، واسم (Name).

٢- يعمل فى الجامعة عدد من أعضاء هيئة التدريس، ولكل عضو هيئة تدريس (Faculty) رقـم (Faculty_ID) يميزه عن بقية أعضاء هيئة التدريس، واسـم (Name) يتكون من (الاسم الأول (FName) واسم العائلة (LName))، وراتب شهرى (Salary)، وتاريخ ميلاد (Date of Birth (DOB))، ورقم هاتف (Phone_No).

٣- يدرس فى الجامعة عدد من الطلاب، ولكل طالب (Student) رقم (Student_ID) يميزه عـن بقية الطلاب فى الجامعة، واسـم (Name) يتكون من (الاسـم الأول (FName) واسم العائلة (LName))، وعنوان بريدى (Address) يتكون من (اسم الشارع (Street)، واسم المدينة (City)، والرمز البريدى (Zip_Code)).

٤- تنفـذ الجامعـة مجموعة من المواد الدراسـية، ولكل مادة دراسـية (Course) رمز (Course_ID) يميزها عن بقية المواد الدراسـية التى تنفذها الجامعة، واسم (Title)، وعدد وحدات (أو ساعات) دراسية (Units).

٥- تنفذ (أو تعقد) كل مادة دراسـية من خلال مجموعة (أو شـعبة) دراسية (Section) واحدة أو أكثر فى الفصل الدراسـى الواحد، أو قد لا تنفذ (أو تعقد) أى مجموعة (أو شـعبة) للمادة الدراسية فى فصل دراسـى معين، ولكل مجموعة دراسية رمز (Section_ID) يتكون من (رقم المجموعة (Section_No)، والفصل الدراسى المنفذة فيه (Semester)، والسنة الدراسية المنفذة فيها (Year)). أما رقم المجموعة (Section_No) فهـو عبارة عن رقم (مثل ١، ٢، ٣، ... إلخ) يميـز المجموعة عن بقية المجموعات المنفذة للمادة الدراسـية نفسـها وفى نفس الفصل والسـنة الدراسيين، ولكنه لا يميزها بشكل منفرد عن بقية المجموعات الدراسية المنفذة للمواد الدراسية الأخرى فى الجامعة.

٦- قد يكون للمادة الدراسية الواحدة مجموعة من المتطلبات الدراسية أو قد لا يكون للمادة الدراسية أى متطلبات دراسية. كما أن المادة الدراسية الواحدة قد تكون متطلباً لأكثر من مادة دراسية أو قد لا تكون متطلباً لأى مادة دراسية.

٧- يعمل (works for) فى كل قسم من أقسام الجامعة عضو هيئة تدريس واحد أو أكثر، وكل عضو من أعضاء هيئة التدريس يعمل فى قسم دراسى واحد فقط.

٨- كل عضو هيئة تدريس فى الجامعة مؤهل (Qualified) لتدريس مادة دراسية واحدة على الأقل، وقد يتوافر للمادة الدراسية الواحدة أكثر من عضو هيئة تدريس مؤهلاً لتدريسها أو قد لا يوجد من أعضاء هيئة التدريس من هو مؤهل لتدريس المادة.

٩- عندما يتأهل عضو هيئة التدريس لتدريس مادة ما لأول مرة، يكون هنالك تاريخ لتأهيله (Qualification date) يحدد تاريخ تأهل عضو هيئة التدريس لتدريس المادة الدراسية.

١٠- تدار (Administered) كل مادة دراسية من قبل قسم دراسى واحد من أقسام الجامعة، ويدير كل قسم مادة دراسية واحدة على الأقل.

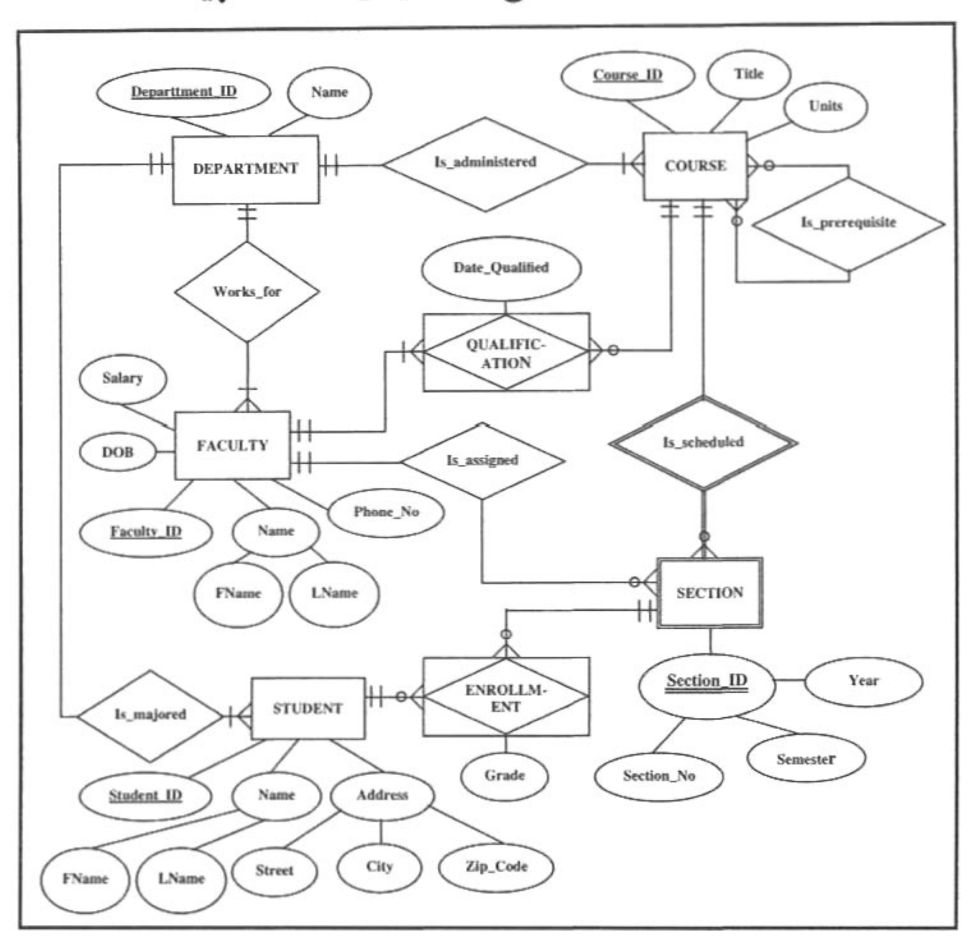
١١- قد يسجل (Enrolls) الطالب الواحد فى أكثر من مجموعة (أو شعبة) دراسية أو قد لا يسجل فى أى مجموعة (أو شعبة) دراسية، والمجموعة (أو الشعبة) الدراسية الواحدة قد لا يسجل فيها أى طالب أو قد يسجل فيها أكثر من طالب.

١٢- عندما يسجل طالب فى مجموعة دراسية تكون له درجة (Grade) تعطى عند انتهائه من الدراسة فى المجموعة.

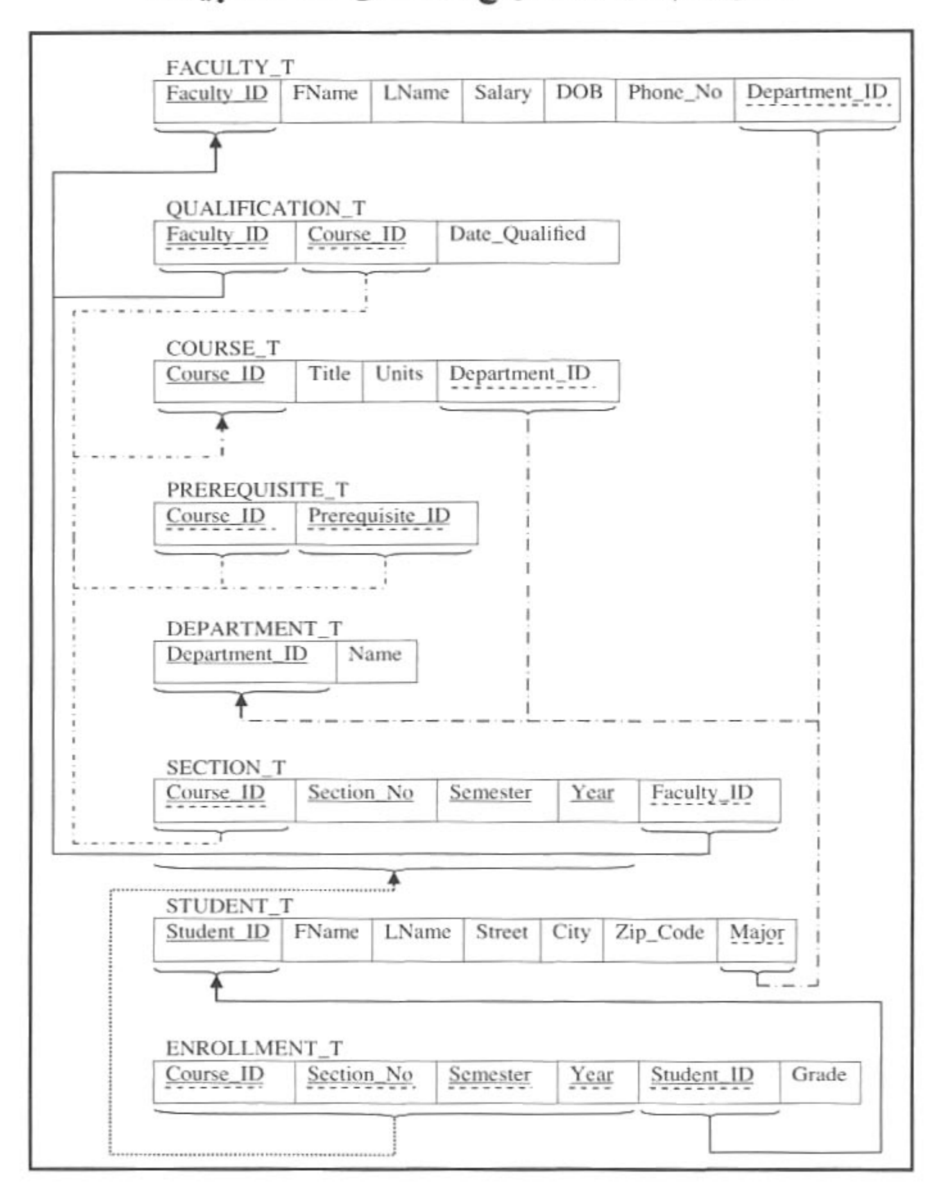
١٣- يتخصص كل طالب (Majors) فى قسم دراسى واحد فقط، ويتخصص فى القسم الدراسى الواحد أكثر من طالب.

١٤- يُكَلَّف (Assigned) كل عضو هيئة تدريس بتدريس مجموعة (أو شعبة) دراسية واحدة أو أكثر وقد لا يكلف عضو هيئة التدريس بأى مجموعة (أو شعبة) دراسية، والمجموعة (أو الشعبة) الدراسية الواحدة تكلف لعضو هيئة تدريس واحد فقط.

ملحق رقم (١)-٢ النموذج المفاهيمى لقاعدة البيانات



ملحق رقم (١)-٣ النموذج المنطقى لقاعدة البيانات



ملحق رقم (١)-٤ جداول قاعدة البيانات حسب بنائها باستخدام نظام إدارة قاعدة بيانات أكسس

- جدول الأقسام العلمية (DEPARTMENT_T):

مواصفات الحقول			
Department_ID	Primary Key	Text(6)	رمز القسم
Name		Text(30)	مسمى القسم

DEPARTMENT_T	
Department_ID	Name
CHEM	Chemistry
CS	Computer Science
EE	Electrical Engineering
ENGL	English Language
MATH	Mathematics
PHYS	Physics
STAT	Statistics

- جدول المواد الدراسية (COURSE_T):

مواصفات الحقول			
Course_ID	Primary Key	Text(7)	رمز المادة الدراسية
Title		Text(35)	مسمى المادة الدراسية
Units		Number(Byte)	عدد وحدات (أو ساعات) المادة الدراسية
Department_ID		Text(6)	رمز القسم الذى تتبعه المادة الدراسية

COURSE_T			
Course_ID	Title	Units	Department_ID
CHEM101	Chemistry (I)	3	CHEM
CHEM102	Chemistry (II)	3	CHEM
CS101	Java Programming	3	CS
CS102	Software Engineering	3	CS
CS103	C/C++ Programming	3	CS
CS104	Computer Architecture	3	CS
CS105	Introduction to Database Systems	3	CS
EE101	Electric Circuits	3	EE
EE102	Electronics (I)	3	EE
EE103	Electronics (II)	3	EE
EE104	Communication Networks	4	EE
ENGL101	English Grammar	2	ENGL
ENGL102	English Writing	3	ENGL
ENGL103	Technical Writing	3	ENGL
MATH101	Introduction to Mathematics	3	MATH
MATH102	Differential Equations	3	MATH
MATH103	Calculus (I)	3	MATH
MATH104	Calculus (II)	3	MATH
MATH106	Algebra	4	MATH
MATH107	Computer Mathematics	3	MATH
PHYS101	Physics (I)	3	PHYS
PHYS102	Physics (II)	3	PHYS
STAT101	Introduction to Statistics	3	STAT
STAT102	Advanced Statistics	3	STAT

- **جدول المواد الدراسية المتطلبة (PREREQUISITE_T):**

مواصفات الحقول			
Course_ID	Primary Key	Text(7)	رمز المادة الدراسية
Prerequisite_ID	Primary Key	Text(7)	رمز المادة الدراسية المتطلبة

PREREQUISITE_T	
Course_ID	**Prerequisite_ID**
CHEM102	CHEM101
CS102	MATH101
CS103	CS102
CS105	MATH101
EE102	EE101
EE103	EE102
EE103	MATH101
MATH102	MATH101
MATH103	MATH101
MATH104	MATH103
MATH106	MATH101
MATH107	MATH101
PHYS102	PHYS101
STAT102	STAT101

- جدول أعضاء هيئة التدريس (FACULTY_T):

مواصفات الحقول			
Faculty_ID	Primary Key	Text(8)	هوية عضو هيئة التدريس
FName		Text(12)	الاسم الأول لعضو هيئة التدريس
LName		Text(12)	اسم عائلته
Phone_No		Text(8)	رقم الهاتف
Salary		Number(Long)	راتبه الشهرى
DOB		Date/Time(Short)	تاريخ ميلاده
Department_ID		Text(6)	رمز القسم الذى يتبعه العضو

FACULTY_T						
Faculty_ID	FName	LName	Phone_NO	Salary	DOB	Department_ID
200	Khalid	Aloufi	454-2341	35000	22/05/1963	MATH
220	Fahad	Alhamid	456-7733	25900	07/10/1970	MATH
310	Saleh	Aleesa	454-8932	30000	13/09/1966	CS
320	Mohammed	Alhamad	454-5412	44000	13/05/1965	CS
330	Ghanim	Alghanim	456-2234	44500	12/08/1969	CS
340	Ibraheem	Alsaleh	454-1234	25000	20/01/1970	CS
400	Ahmad	Alotaibi	454-4563	33900	17/05/1971	CHEM
420	Saleh	Alghamdi	454-2233	44600	13/02/1969	CHEM
500	Yahya	Khorshid	456-2221	36700	12/03/1965	ENGL
540	Salem	Alhamad	456-3304	40000	11/09/1972	ENGL
560	Salman	Albassam	454-7865	33800	13/09/1968	ENGL
600	Turki	Alturki	456-7891	27800	23/07/1975	STAT
640	Fahad	Alzaid	456-3322	44300	12/05/1971	STAT
660	Saud	Alkhalifa	454-9856	44900	13/08/1972	STAT
710	Mahmood	Alsalem	456-3323	31900	19/02/1973	PHYS
730	Mishal	Almazid	454-2343	29800	17/09/1975	PHYS
770	Sultan	Aljasir	456-3212	43300	13/05/1970	PHYS
800	Ali	Albader	456-7812	45300	22/06/1966	EE
810	Saad	Alzhrani	454-5578	44200	17/10/1967	EE
850	Ahmad	Alsabti	456-0120	33900	15/04/1973	EE

- جدول القدرات التعليمية (QUALIFICATION_T):

مواصفات الحقول			
Course_ID	Primary Key	Text(7)	رمز المادة الدراسية
Faculty_ID	Primary Key	Text(8)	هوية عضو هيئة التدريس
Date_Qualified		Date/Time(Short)	تاريخ التأهيل لتدريس المادة

QUALIFICATION_T		
Couse_ID	Faculty_ID	Date_Qualified
CHEM101	400	02/01/1991
CHEM102	420	02/07/1992
CS101	310	05/06/1995
CS102	320	09/08/1995
CS103	320	03/09/1996
CS104	330	02/10/1997
CS105	340	02/12/1997
EE101	800	08/01/1993
EE102	810	12/03/1994
EE103	850	15/11/1995
EE104	810	03/02/1996
ENGL101	500	01/07/1995
ENGL102	540	02/08/1994
ENGL103	560	09/09/1993
MATH101	200	13/11/1991
MATH102	200	02/06/1993
MATH103	220	02/07/1993
MATH104	220	13/08/1993
MATH106	220	17/10/1994
MATH107	200	10/01/1995
PHYS101	710	13/07/1996
PHYS101	770	11/02/1996
PHYS102	730	02/01/1997
STAT101	600	15/08/1993
STAT101	660	03/04/1995
STAT102	640	02/05/1994

- جدول المجموعات (أو الشعب) الدراسية (SECTION_T):

مواصفات الحقول			
Course_ID	Primary Key	Text(7)	رمز المادة الدراسية
Section_No	Primary Key	Number(Byte)	رقم المجموعة (أو الشعبة)
Semester	Primary Key	Text(10)	الفصل الدراسى المنفذة فيه
Year	Primary Key	Number(Integer)	السنة الدراسية المنفذة فيها
Faculty_ID		Text(8)	هوية عضو هيئة التدريس

SECTION_T

Course_ID	Section_No	Semester	Year	Faculty_ID
CHEM101	1	FALL	2000	400
CHEM101	2	FALL	2000	400
CS101	1	FALL	2000	310
CS101	2	FALL	2000	310
CS102	1	SPRING	2000	320
CS103	1	SPRING	2000	320
CS104	1	FALL	2001	330
CS105	1	SPRING	2001	340
EE101	1	FALL	2001	800
EE102	1	SPRING	2001	810
ENGL101	1	FALL	2000	500
ENGL102	1	SPRING	2000	540
MATH101	1	FALL	2000	200
MATH102	1	SPRING	2000	200
MATH103	1	FALL	2001	220
MATH104	1	SPRING	2001	220
PHYS101	1	FALL	2001	710
PHYS102	1	SPRING	2001	730
STAT101	1	SPRING	2000	600
STAT102	1	SPRING	2001	640

- جدول الطلاب (STUDENT_T):

مواصفات الحقول			
Student_ID	Primary Key	Text(8)	هوية الطالب
FName		Text(12)	الاسم الأول للطالب
LName		Text(12)	اسم العائلة
Street		Text(30)	اسم الشارع الذى يسكن فيه الطالب
City		Text(8)	اسم المدينة التى يسكنها الطالب
Zip_Code		Text(5)	الرمز البريدى
Major		Text(6)	رمز القسم الذى يتبعه الطالب

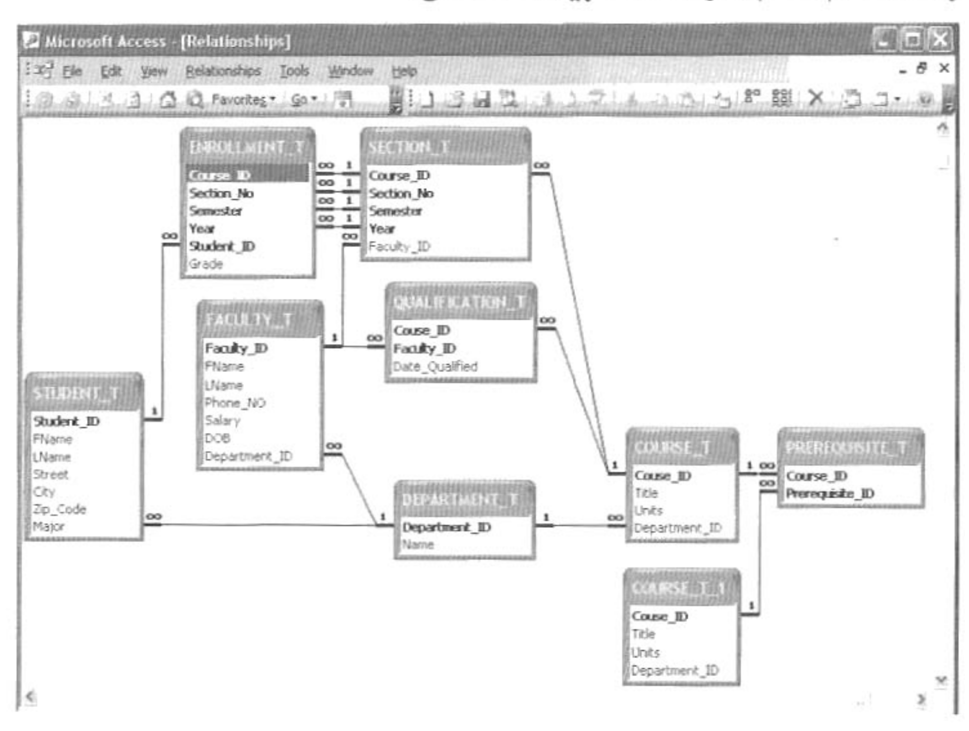
STUDENT_T						
Student_ID	**FName**	**LName**	**Street**	**City**	**Zip_Code**	**Major**
19992020	Saleh	Alhamad	13 Almutanabi Street	Riyadh	11121	CS
19992341	Abdullah	Aloufi	25 Jareer Street	Riyadh	12123	CHEM
19994512	Salem	Algamdi	98 Bin Taimiah Street	Jeddah	34565	PHYS
20001111	Mishal	Alyousef	13 Alsouk Street	Taif	67156	CS
20001212	Khalid	Alsultan	22 Bin Hamdan Street	Jeddah	34565	MATH
20001213	Mohammed	Abdelaleem	10 Bin Hamdan Street	Jeddah	35787	STAT
20001214	Sami	Aloutaibi	67 Alfadel Street	Dammam	26123	ENGL
20001215	Saud	Alganim	24 Alfadel Street	Dammam	27145	EE
20011212	Abdulrahman	Abdulsalam	10 Almadinah Street	Skaka	88756	CHEM
20011213	Salman	Alsaleh	15 King Fahad Road	Dammam	28898	PHYS
20011214	Khalid	Alomar	91 Alwadi Street	Najran	90987	MATH
20011215	Minwer	Almutairi	87 Alhamra Road	Jizan	92347	STAT
20011216	Turki	Alassaf	25 Prince Abdullah Street	Riyadh	11897	ENGL
20011217	Saleh	Alzaid	25 King Faisal Street	Riyadh	11874	EE
20021111	Ghanim	Alhmoud	56 Altahliah Street	Jeddah	35234	CS
20021212	Sultan	Abdulgader	123 Salman Alfarsi Street	Riyadh	12657	CHEM
20021213	Suliman	Almushari	45 Prince Sultan Street	Najran	90888	PHYS
20021214	Ahmad	Alsaif	13 Khalifa Street	Taif	67898	MATH
20021234	Ahmad	Alshemamri	15 Othman street	Jizan	92534	ENGL
20022345	Mohammed	Alzamil	67 Abubaker Road	Abha	56879	STAT
20023678	Mansour	Alzamil	13 King Abdulaziz Road	Tabouk	78453	EE

- جدول تسجيل الطلبة (ENROLLMENT_T):

مواصفات الحقول			
Course_ID	Primary Key	Text(7)	رمز المادة الدراسية
Section_No	Primary Key	Number(Byte)	رقم المجموعة (أو الشعبة)
Semester	Primary Key	Text(10)	الفصل الدراسى المنفذة فيه
Year	Primary Key	Number(Integer)	السنة الدراسية المنفذة فيها
Student_ID	Primary Key	Text(8)	هوية الطالب المسجل فى المجموعة
Grade		Number(Byte)	درجة الطالب فى المادة

ENROLLMENT_T					
Course_ID	Section_No	Semester	Year	Student_ID	Grade
CHEM101	1	FALL	2000	19992020	4
CHEM101	1	FALL	2000	19992341	3
CHEM101	1	FALL	2000	20001212	4
CHEM101	2	FALL	2000	19994512	3
CHEM101	2	FALL	2000	20001111	1
CS101	1	FALL	2000	19992020	2
CS101	2	FALL	2000	20001111	4
CS102	1	SPRING	2000	19992020	3
CS102	1	SPRING	2000	20001111	4
ENGL101	1	FALL	2000	19992020	3
ENGL101	1	FALL	2000	19992341	4
ENGL101	1	FALL	2000	19994512	4
ENGL101	1	FALL	2000	20001111	4
ENGL102	1	SPRING	2000	19992020	1
ENGL102	1	SPRING	2000	20001111	4
MATH101	1	FALL	2000	19992020	3
MATH101	1	FALL	2000	19992341	2
MATH101	1	FALL	2000	19994512	0
MATH101	1	FALL	2000	20001111	2
MATH102	1	SPRING	2000	19992020	2
MATH102	1	SPRING	2000	20001111	0
STAT101	1	SPRING	2000	19992020	2
STAT101	1	SPRING	2000	20001111	3

ملحق رقم (١)-٥ العلاقات بين جداول قاعدة البيانات حسب بنائها باستخدام نظام إدارة قاعدة بيانات أكسس:



ملحق رقم(١)-٦ إنشاء قاعدة البيانات باستخدام تعليمات (SQL) فى بيئة أوراكل (SQL*Plus):

عند إنشاء جداول قاعدة بيانات الجمعية الأهلية أو أية قاعدة بيانات أخرى، وعلى خلاف ما هو متبع فى بيئة نظام قاعدة بيانات أكسس، يجب ملاحظة ترتيب إنشاء الجداول بحيث يتم إنشاء الجداول التى تحتوى على مفاتيح خارجية (لتمثيل العلاقات) بعد إنشاء الجداول التى تحتوى على المفاتيح الرئيسية التى تربطها بالمفاتيح الخارجية. وفى حالة عدم اتباع ذلك، فإن نظام إدارة قاعدة البيانات لن يقوم بإنشاء هذه الجداول وسيصدر عنه خطأ نتيجة لمثل هذا الإجراء. ويعزى السبب فى ذلك، فى هذه الحالة، إلى أن المستخدم يحاول إنشاء جدول يحتوى على حقل (أو حقول) ترتبط بحقول أخرى (وهى المفاتيح الرئيسية)، وهذه الحقول غير معرفة أصلاً فى قاعدة البيانات. وفى حالة عدم رغبة المستخدم فى إنشاء الجداول، وفق ترتيب معين، أو فى حالة وجود علاقات (أو قيود) متداخلة (Cyclic Rrelationships)؛ فإنه يمكن إنشاء الجداول دون إنشاء العلاقات (أو القيود) فيما بينها. وبعد إنشاء الجداول تتم إضافة العلاقات (أو القيود) باستخدام عبارة «تعديل جدول» (ALTER TABLE) (Garcia-Molina et al, 2002).

أما فى حالة نظام إدارة قاعدة بيانات أكسس، فإنه يتم تمثيل العلاقات (أو القيود) ما بين جداول قاعدة البيانات فى مرحلة تلى مرحلة إنشاء الجداول، ومن ثم لا تظهر مثل هذه المشكلة للمستخدم عند إنشائه للجداول والعلاقات فيما بينها.

- جدول الأقسام العلمية (DEPARTMENT_T):

إنشاء جدول الأقسام العلمية

```
CREATE TABLE DEPARTMENT_T
        (DEPARTMENT_ID      CHAR(6)      NOT NULL,
         NAME               CHAR(30)     NOT NULL,
CONSTRAINT DEPARTMENT_PK PRIMARY KEY (DEPARTMENT_ID));
```

ويتم إدخال البيانات للجدول باستخدام (SQL) حسب التعليمات التالية:

إدخال بيانات جدول الأقسام العلمية

```
INSERT INTO DEPARTMENT_T VALUES ('CHEM', 'Chemistry');
INSERT INTO DEPARTMENT_T VALUES ('CS', 'Computer Science');
INSERT INTO DEPARTMENT_T VALUES ('EE', 'Electrical Engineering');
INSERT INTO DEPARTMENT_T VALUES ('ENGL', 'English Language');
INSERT INTO DEPARTMENT_T VALUES ('MATH', 'Mathematics');
INSERT INTO DEPARTMENT_T VALUES ('PHYS', 'Physics');
INSERT INTO DEPARTMENT_T VALUES ('STAT', 'Statistics');
```

- جدول المواد الدراسية (COURSE_T):

إنشاء جدول المواد الدراسية

```
CREATE TABLE COURSE_T
        (COURSE_ID          CHAR(7)      NOT NULL,
         TITLE              CHAR(35)     NOT NULL,
         UNITS              NUMBER       NOT NULL,
         DEPARTMENT_ID      CHAR(6)      NOT NULL,
CONSTRAINT COURSE_PK  PRIMARY KEY (COURSE_ID),
CONSTRAINT COURSE_FK1  FOREIGN KEY (DEPARTMENT_ID)
            REFERENCES DEPARTMENT_T(DEPARTMENT_ID));
```

إدخال بيانات المواد الدراسية

```
INSERT INTO COURSE_T VALUES
        ('CHEM101', 'CHEMISTRY (I)', 3, 'CHEM');
INSERT INTO COURSE_T VALUES
        ('CHEM102', 'CHEMISTRY (II)', 3, 'CHEM');
INSERT INTO COURSE_T VALUES
        ('CS101', 'JAVA PROGRAMMING', 3, 'CS');
INSERT INTO COURSE_T VALUES
        ('CS102', 'SOFTWARE ENGINEERING', 3, 'CS');
INSERT INTO COURSE_T VALUES
        ('CS103', 'C/C++ PROGRAMMING', 3, 'CS');
INSERT INTO COURSE_T VALUES
        ('CS104', 'COMPUTER ARCHITECTURE', 3, 'CS');
INSERT INTO COURSE_T VALUES
        ('CS105', 'INTRODUCTION TO DATABASE SYSTEMS', 3, 'CS');
INSERT INTO COURSE_T VALUES
        ('EE101', 'ELECTRIC CIRCUITS', 3, 'EE');
INSERT INTO COURSE_T VALUES
        ('EE102', 'ELECTRONICS (I)', 3, 'EE');
INSERT INTO COURSE_T VALUES
        ('EE103', 'ELECTRONICS (II)', 3, 'EE');
INSERT INTO COURSE_T VALUES
        ('EE104', 'COMMUNICATION NETWORKS', 4, 'EE');
INSERT INTO COURSE_T VALUES
        ('ENGL101', 'ENGLISH GRAMMAR', 2, 'ENGL');
INSERT INTO COURSE_T VALUES
        ('ENGL102', 'ENGLISH WRITING', 3, 'ENGL');
INSERT INTO COURSE_T VALUES
        ('ENGL103', 'TECHNICAL WRITING', 3, 'ENGL');
INSERT INTO COURSE_T VALUES
        ('MATH101', 'INTRODUCTION To MATHEMATICS', 3, 'MATH');
INSERT INTO COURSE_T VALUES
        ('MATH102', 'DIFFERENTIAL EQUATIONS', 3, 'MATH');
INSERT INTO COURSE_T VALUES
        ('MATH103', 'CALCULUS (I)', 3, 'MATH');
INSERT INTO COURSE_T VALUES
        ('MATH104', 'CALCULUS (II)', 3, 'MATH');
INSERT INTO COURSE_T VALUES
        ('MATH106', 'ALGEBRA', 4, 'MATH');
INSERT INTO COURSE_T VALUES
        ('MATH107', 'COMPUTER MATHEMATICS', 3, 'MATH');
INSERT INTO COURSE_T VALUES
        ('PHYS101', 'PHYSICS (I)', 3, 'PHYS');
INSERT INTO COURSE_T VALUES
        ('PHYS102', 'PHYSICS (II)', 3, 'PHYS');
INSERT INTO COURSE_T VALUES
        ('STAT101', 'INTRODUCTION TO STATISTICS', 3, 'STAT');
INSERT INTO COURSE_T VALUES
        ('STAT102', 'ADVANCED STATISTICS', 3, 'STAT');
```

- جدول المواد الدراسية المتطلبة (PREREQUISITE_T):

إنشاء جدول المواد الدراسية المتطلبة

```
CREATE TABLE PREREQUISITE_T
        (COURSE_ID              CHAR(7)         NOT NULL,
         PREREQUISITE_ID        CHAR(7)         NOT NULL,
CONSTRAINT RREREQUISITE_PK  PRIMARY KEY (COURSE_ID,
PREREQUISITE_ID),
CONSTRAINT PREREQUISITE_FK1 FOREIGN KEY (COURSE_ID)
                                REFERENCES COURSE_T(COURSE_ID),
CONSTRAINT PREREQUISITE_FK2 FOREIGN KEY (PREREQUISITE_ID)
                                REFERENCES COURSE_T(COURSE_ID));
```

إدخال بيانات المواد الدراسية المتطلبة

```
INSERT INTO PREREQUISITE_T VALUES ('CHEM102', 'CHEM101');
INSERT INTO PREREQUISITE_T VALUES ('CS102', 'MATH101');
INSERT INTO PREREQUISITE_T VALUES ('CS103', 'CS102');
INSERT INTO PREREQUISITE_T VALUES ('CS105', 'MATH101');
INSERT INTO PREREQUISITE_T VALUES ('EE102', 'EE101');
INSERT INTO PREREQUISITE_T VALUES ('EE103', 'EE102');
INSERT INTO PREREQUISITE_T VALUES ('EE103', 'MATH101');
INSERT INTO PREREQUISITE_T VALUES ('MATH102', 'MATH101');
INSERT INTO PREREQUISITE_T VALUES ('MATH103', 'MATH101');
INSERT INTO PREREQUISITE_T VALUES ('MATH104', 'MATH103');
INSERT INTO PREREQUISITE_T VALUES ('MATH106', 'MATH101');
INSERT INTO PREREQUISITE_T VALUES ('MATH107', 'MATH101');
INSERT INTO PREREQUISITE_T VALUES ('PHYS102', 'PHYS101');
INSERT INTO PREREQUISITE_T VALUES ('STAT102', 'STAT101');
```

- جدول أعضاء هيئة التدريس (FUCULTY_T):

إنشاء جدول أعضاء هيئة التدريس

```
CREATE TABLE FACULTY_T
    (FACULTY_ID         CHAR(8)         NOT NULL,
     FNAME              CHAR(12)        NOT NULL,
     LNAME              CHAR(12)        NOT NULL,
     PHONE_NO           CHAR(8),
     SALARY             NUMBER(9,2),
     DOB                DATE            NOT NULL,
     DEPARTMENT_ID      CHAR(6)         NOT NULL,
CONSTRAINT FACULTY_PK  PRIMARY KEY (FACULTY_ID),
CONSTRAINT FACULTY_FK1 FOREIGN KEY (DEPARTMENT_ID)
                REFERENCES DEPARTMENT_T(DEPARTMENT_ID));
```

إدخال بيانات أعضاء هيئة التدريس

```
INSERT INTO FACULTY_T VALUES
        ('200', 'Khalid', 'Aloufi', '454-2341', 35000, '22-MAY-1963', 'MATH');
INSERT INTO FACULTY_T VALUES
        ('220', 'Fahad', 'Alhamid', '456-7733', 25900, '07-OCT-1970', 'MATH');
INSERT INTO FACULTY_T VALUES
        ('310', 'Saleh', 'Aleesa', '454-8932', 30000, '13-SEP-1966', 'CS');
INSERT INTO FACULTY_T VALUES
        ('320', 'Mohammed', 'Alhamad', '454-5412', 44000, '13-MAY-1965', 'CS');
INSERT INTO FACULTY_T VALUES
        ('330', 'Ghanim', 'Alghanim', '456-2234', 44500, '12-AUG-1969', 'CS');
INSERT INTO FACULTY_T VALUES
        ('340', 'Ibraheem', 'Alsaleh', '454-1234', 25000, '20-JAN-1970', 'CS');
INSERT INTO FACULTY_T VALUES
        ('400', 'Ahmad', 'Alotaibi', '454-4563', 33900, '17-MAY-1971', 'CHEM');
INSERT INTO FACULTY_T VALUES
        ('420', 'Saleh', 'Alghamdi', '454-2233', 44600, '13-FEB-1969', 'CHEM');
INSERT INTO FACULTY_T VALUES
        ('500', 'Yahya', 'Khorshid', '456-2221', 36700, '12-MAR-1965', 'ENGL');
INSERT INTO FACULTY_T VALUES
        ('540', 'Salem', 'Alhamad', '456-3304', 40000, '11-SEP-1972', 'ENGL');
INSERT INTO FACULTY_T VALUES
        ('560', 'Salman', 'Albassam', '454-7865', 33800, '13-SEP-1968', 'ENGL');
INSERT INTO FACULTY_T VALUES
        ('600', 'Turki', 'Alturki', '456-7891', 27800, '23-JUL-1975', 'STAT');
INSERT INTO FACULTY_T VALUES
        ('640', 'Fahad', 'Alzaid', '456-3322', 44300, '12-MAY-1971', 'STAT');
INSERT INTO FACULTY_T VALUES
        ('660', 'Saud', 'Alkhalifa', '454-9856', 44900, '13-AUG-1972', 'STAT');
INSERT INTO FACULTY_T VALUES
        ('710', 'Mahmood', 'Alsalem', '456-3323', 31900, '19-FEB-1973', 'PHYS');
INSERT INTO FACULTY_T VALUES
        ('730', 'Mishal', 'Almazid', '454-2343', 29800, '17-SEP-1975', 'PHYS');
INSERT INTO FACULTY_T VALUES
        ('770', 'Sultan', 'Aljasir', '456-3212', 43300, '13-MAY-1970', 'PHYS');
INSERT INTO FACULTY_T VALUES
        ('800', 'Ali', 'Albader', '456-7812', 45300, '22-JUN-1966', 'EE');
INSERT INTO FACULTY_T VALUES
        ('810', 'Saad', 'Alzhrani', '454-5578', 44200, '17-OCT-1967', 'EE');
INSERT INTO FACULTY_T VALUES
        ('850', 'Ahmad', 'Alsabti', '456-0120', 33900, '15-APR-1973', 'EE');
```

- جدول القدرات التعليمية لأعضاء هيئة التدريس (QUALIFICATION_T):

إنشاء جدول المؤهلات التعليمية لأعضاء هيئة التدريس

```
CREATE TABLE QUALIFICATION_T
        (COURSE_ID          CHAR(7)          NOT NULL,
         FACULTY_ID         CHAR(8)          NOT NULL,
         DATE_QUALIFIED     DATE             NOT NULL,
CONSTRAINT QUALIFICATION_PK   PRIMARY KEY (COURSE_ID, FACULTY_ID),
CONSTRAINT QUALIFICATION_FK1 FOREIGN KEY (COURSE_ID)
                              REFERENCES COURSE_T(COURSE_ID),
CONSTRAINT QUALIFICATION_FK2 FOREIGN KEY (FACULTY_ID)
                              REFERENCES FACULTY_T(FACULTY_ID));
```

إدخال بيانات المؤهلات التعليمية لأعضاء هيئة التدريس

```
INSERT INTO QUALIFICATION_T VALUES ('CHEM101', 400, '02-JAN-1991');
INSERT INTO QUALIFICATION_T VALUES ('CHEM102', 420, '02-JUL-1992');
INSERT INTO QUALIFICATION_T VALUES ('CS101', 310, '05-JUN-1995');
INSERT INTO QUALIFICATION_T VALUES ('CS102', 320, '09-AUG-1995');
INSERT INTO QUALIFICATION_T VALUES ('CS103', 320, '03-AUG-1996');
INSERT INTO QUALIFICATION_T VALUES ('CS104', 330, '02-SEP-1997');
INSERT INTO QUALIFICATION_T VALUES ('CS105', 340, '02-DEC-1997');
INSERT INTO QUALIFICATION_T VALUES ('EE101', 800, '08-JAN-1993');
INSERT INTO QUALIFICATION_T VALUES ('EE102', 810, '12-MAR-1994');
INSERT INTO QUALIFICATION_T VALUES ('EE103', 850, '15-NOV-1995');
INSERT INTO QUALIFICATION_T VALUES ('EE104', 810, '03-JAN-1996');
INSERT INTO QUALIFICATION_T VALUES ('ENGL101', 500, '01-JUL-1995');
INSERT INTO QUALIFICATION_T VALUES ('ENGL102', 540, '02-AUG-1994');
INSERT INTO QUALIFICATION_T VALUES ('ENGL103', 560, '09-SEP-1993');
INSERT INTO QUALIFICATION_T VALUES ('MATH101', 200, '13-NOV-1991');
INSERT INTO QUALIFICATION_T VALUES ('MATH102', 200, '02-JUN-1993');
INSERT INTO QUALIFICATION_T VALUES ('MATH103', 220, '02-JUL-1993');
INSERT INTO QUALIFICATION_T VALUES ('MATH104', 220, '13-AUG-1993');
INSERT INTO QUALIFICATION_T VALUES ('MATH106', 220, '17-OCT-1994');
INSERT INTO QUALIFICATION_T VALUES ('MATH107', 200, '10-JAN-1995');
INSERT INTO QUALIFICATION_T VALUES ('PHYS101', 710, '13-JUL-1996');
INSERT INTO QUALIFICATION_T VALUES ('PHYS101', 770, '11-FEB-1996');
INSERT INTO QUALIFICATION_T VALUES ('PHYS102', 730, '02-JAN-1997');
INSERT INTO QUALIFICATION_T VALUES ('STAT101', 600, '15-AUG-1993');
INSERT INTO QUALIFICATION_T VALUES ('STAT101', 660, '03-APR-1995');
INSERT INTO QUALIFICATION_T VALUES ('STAT102', 640, '02-MAY-1994');
```

- جدول المجموعات (أو الشعب) الدراسية (SECTION_T):

إنشاء جدول المجموعات (أو الشعب) الدراسية

```
CREATE TABLE SECTION_T
        (COURSE_ID      CHAR(7)        NOT NULL,
         SECTION_NO     NUMBER         NOT NULL,
         SEMESTER       CHAR(10)       NOT NULL,
         YEAR           NUMBER         NOT NULL,
         FACULTY_ID     CHAR(8)        NOT NULL,
CONSTRAINT SECTION_PK   PRIMARY KEY (COURSE_ID, SECTION_NO,
SEMESTER, YEAR),
CONSTRAINT SECTION_FK1 FOREIGN KEY (COURSE_ID)
                       REFERENCES COURSE_T(COURSE_ID),
CONSTRAINT SECTION_FK2 FOREIGN KEY (FACULTY_ID)
                      REFERENCES FACULTY_T(FACULTY_ID));
```

إدخال بيانات المجموعات (أو الشعب) الدراسية

```
INSERT INTO SECTION_T VALUES ('CHEM101', 1, 'FALL', 2000, 400);
INSERT INTO SECTION_T VALUES ('CHEM101', 2, 'FALL', 2000, 400);
INSERT INTO SECTION_T VALUES ('CS101', 1, 'FALL', 2000, 310);
INSERT INTO SECTION_T VALUES ('CS101', 2, 'FALL', 2000, 310);
INSERT INTO SECTION_T VALUES ('CS102', 1, 'SPRING', 2000, 320);
INSERT INTO SECTION_T VALUES ('CS103', 1, 'SPRING', 2000, 320);
INSERT INTO SECTION_T VALUES ('CS104', 1, 'FALL', 2001, 330);
INSERT INTO SECTION_T VALUES ('CS105', 1, 'SPRING', 2001, 340);
INSERT INTO SECTION_T VALUES ('EE101', 1, 'FALL', 2001, 800);
INSERT INTO SECTION_T VALUES ('EE102', 1, 'SPRING', 2001, 810);
INSERT INTO SECTION_T VALUES ('ENGL101', 1, 'FALL', 2000, 500);
INSERT INTO SECTION_T VALUES ('ENGL102', 1, 'SPRING', 2000, 540);
INSERT INTO SECTION_T VALUES ('MATH101', 1, 'FALL', 2000, 200);
INSERT INTO SECTION_T VALUES ('MATH102', 1, 'SPRING', 2000, 200);
INSERT INTO SECTION_T VALUES ('MATH103', 1, 'FALL', 2001, 220);
INSERT INTO SECTION_T VALUES ('MATH104', 1, 'SPRING', 2001, 220);
INSERT INTO SECTION_T VALUES ('PHYS101', 1, 'FALL', 2001, 710);
INSERT INTO SECTION_T VALUES ('PHYS102', 1, 'SPRING', 2001, 730);
INSERT INTO SECTION_T VALUES ('STAT101', 1, 'SPRING', 2000, 600);
INSERT INTO SECTION_T VALUES ('STAT102', 1, 'SPRING', 2001, 640);
```

- جدول الطلاب (STUDENT_T):

إنشاء جدول الطلاب

```
CREATE TABLE STUDENT_T
        (STUDENT_ID         CHAR(8)       NOT NULL,
         FNAME              CHAR(12)      NOT NULL,
         LNAME              CHAR(12)      NOT NULL,
         STREET             CHAR(30),
         CITY               CHAR(8),
         ZIP_CODE           CHAR(5),
         MAJOR              CHAR(6)       NOT NULL,
CONSTRAINT STUDENT_PK  PRIMARY KEY (STUDENT_ID),
CONSTRAINT STUDENT_FK1 FOREIGN KEY (MAJOR)
                   REFERENCES DEPARTMENT_T(DEPARTMENT_ID));
```

إدخال بيانات الطلاب

```
INSERT INTO STUDENT_T VALUES
        ('19992020', 'Saleh', 'Alhamad', '13 Almutanabi Street', 'Riyadh', '11121', 'CS');
INSERT INTO STUDENT_T VALUES
        ('19992341', 'Abdullah', 'Aloufi', '25 Jareer Street', 'Riyadh', '12123', 'CHEM');
INSERT INTO STUDENT_T VALUES
        ('19994512', 'Salem', 'Algamdi', '98 Bin Taimiah Street', 'Jeddah', '34565', 'PHYS');
INSERT INTO STUDENT_T VALUES
        ('20001111', 'Mishal', 'Alyousef', '13 Alsouk Street', 'Taif', '67156', 'CS');
INSERT INTO STUDENT_T VALUES
        ('20001212', 'Khalid', 'Alsultan', '22 Bin Hamdan Street', 'Jeddah', '34565', 'MATH');
INSERT INTO STUDENT_T VALUES
        ('20001213', 'Mohammed', 'Abdelaleem', '10 Bin Hamdan Street', 'Jeddah', '35787', 'STAT');
INSERT INTO STUDENT_T VALUES
        ('20001214', 'Sami', 'Aloutaibi', '67 Alfadel Street', 'Dammam', '26123', 'ENGL');
INSERT INTO STUDENT_T VALUES
        ('20001215', 'Saud', 'Alganim', '24 Alfadel Street', 'Dammam', '27145', 'EE');
INSERT INTO STUDENT_T VALUES
        ('20011212', 'Abdulrahman', 'Abdulsalam', '10 Almadinah Street', 'Skaka', '88756', 'CHEM');
INSERT INTO STUDENT_T VALUES
        ('20011213', 'Salman', 'Alsaleh', '15 King Fahad Road', 'Dammam', '28898', 'PHYS');
INSERT INTO STUDENT_T VALUES
        ('20011214', 'Khalid', 'Alomar', '91 Alwadi Street', 'Najran', '90987', 'MATH');
INSERT INTO STUDENT_T VALUES
        ('20011215', 'Minwer', 'Almutairi', '87 Alhamra Road', 'Jizan', '92347', 'STAT');
INSERT INTO STUDENT_T VALUES
        ('20011216', 'Turki', 'Alassaf', '25 Prince Abdullah Street', 'Riyadh', '11897', 'ENGL');
INSERT INTO STUDENT_T VALUES
        ('20011217', 'Saleh', 'Alzaid', '25 King Faisal Street', 'Riyadh', '11874', 'EE');
INSERT INTO STUDENT_T VALUES
        ('20021111', 'Ghanim', 'Alhmoud', '56 Altahliah Street', 'Jeddah', '35234', 'CS');
INSERT INTO STUDENT_T VALUES
        ('20021212', 'Sultan', 'Abdulgader', '123 Salman Alfarsi Street', 'Riyadh', '12657', 'CHEM');
INSERT INTO STUDENT_T VALUES
        ('20021213', 'Suliman', 'Almushari', '45 Prince Sultan Street', 'Najran', '90888', 'PHYS');
INSERT INTO STUDENT_T VALUES
        ('20021214', 'Ahmad', 'Alsaif', '13 Khalifa Street', 'Taif', '67898', 'MATH');
INSERT INTO STUDENT_T VALUES
        ('20021234', 'Ahmad', 'Alshemamri', '15 Othman street', 'Jizan', '92534', 'ENGL');
INSERT INTO STUDENT_T VALUES
        ('20022345', 'Mohammed', 'Alzamil', '67 Abubaker Road', 'Abha', '56879', 'STAT');
INSERT INTO STUDENT_T VALUES
        ('20023678', 'Mansour', 'Alzamil', '13 King Abdulaziz Road', 'Tabouk', '78453', 'EE');
```

- جدول تسجيل الطلبة (ENROLLMENT_T):

إنشاء جدول تسجيل الطلاب

```
CREATE TABLE ENROLLMENT_T
        (COURSE_ID       CHAR(7)         NOT NULL,
        SECTION_NO       NUMBER          NOT NULL,
        SEMESTER         CHAR(10)        NOT NULL,
        YEAR             NUMBER          NOT NULL,
        STUDENT_ID       CHAR(8)         NOT NULL,
        GRADE            NUMBER,
CONSTRAINT ENROLLMENT_PK  PRIMARY KEY
        (COURSE_ID, SECTION_NO, SEMESTER, YEAR, STUDENT_ID),
CONSTRAINT ENROLLMENT_FK1 FOREIGN KEY (COURSE_ID,
SECTION_NO, SEMESTER, YEAR)
      REFERENCES SECTION_T(COURSE_ID, SECTION_NO, SEMESTER, YEAR),
CONSTRAINT ENROLLMENT_FK2 FOREIGN KEY (STUDENT_ID)
          REFERENCES STUDENT_T(STUDENT_ID));
```

إدخال بيانات تسجيل الطلبة

```
INSERT INTO ENROLLMENT_T VALUES ('CHEM101', 1, 'FALL', 2000, '19992020', 4);
INSERT INTO ENROLLMENT_T VALUES ('CHEM101', 1, 'FALL', 2000, '19992341', 3);
INSERT INTO ENROLLMENT_T VALUES ('CHEM101', 1, 'FALL', 2000, '20001212', 4);
INSERT INTO ENROLLMENT_T VALUES ('CHEM101', 2, 'FALL', 2000, '19994512', 3);
INSERT INTO ENROLLMENT_T VALUES ('CHEM101', 2, 'FALL', 2000, '20001111', 1);
INSERT INTO ENROLLMENT_T VALUES ('CS101', 1, 'FALL', 2000, '19992020', 2);
INSERT INTO ENROLLMENT_T VALUES ('CS101', 2, 'FALL', 2000, '20001111', 4);
INSERT INTO ENROLLMENT_T VALUES ('CS102', 1, 'SPRING', 2000, '19992020', 3);
INSERT INTO ENROLLMENT_T VALUES ('CS102', 1, 'SPRING', 2000, '20001111', 4);
INSERT INTO ENROLLMENT_T VALUES ('ENGL101', 1, 'FALL', 2000, '19992020', 3);
INSERT INTO ENROLLMENT_T VALUES ('ENGL101', 1, 'FALL', 2000, '19992341', 4);
INSERT INTO ENROLLMENT_T VALUES ('ENGL101', 1, 'FALL', 2000, '19994512', 4);
INSERT INTO ENROLLMENT_T VALUES ('ENGL101', 1, 'FALL', 2000, '20001111', 4);
INSERT INTO ENROLLMENT_T VALUES ('ENGL102', 1, 'SPRING', 2000, '19992020', 1);
INSERT INTO ENROLLMENT_T VALUES ('ENGL102', 1, 'SPRING', 2000, '20001111', 4);
INSERT INTO ENROLLMENT_T VALUES ('MATH101', 1, 'FALL', 2000, '19992020', 3);
INSERT INTO ENROLLMENT_T VALUES ('MATH101', 1, 'FALL', 2000, '19992341', 2);
INSERT INTO ENROLLMENT_T VALUES ('MATH101', 1, 'FALL', 2000, '19994512', 0);
INSERT INTO ENROLLMENT_T VALUES ('MATH101', 1, 'FALL', 2000, '20001111', 2);
INSERT INTO ENROLLMENT_T VALUES ('MATH102', 1, 'SPRING', 2000, '19992020', 2);
INSERT INTO ENROLLMENT_T VALUES ('MATH102', 1, 'SPRING', 2000, '20001111', 0);
INSERT INTO ENROLLMENT_T VALUES ('STAT101', 1, 'SPRING', 2000, '19992020', 2);
INSERT INTO ENROLLMENT_T VALUES ('STAT101', 1, 'SPRING', 2000, '20001111', 3);
```

ملحق رقم (١)-٧ استعراض لمحتويات جداول قاعدة البيانات بعد إنشائها فى بيئة أوراكل باستخدام تعليمة (SELECT * FROM TableName)

- جدول الأقسام العلمية (DEPARTMENT_T):

DEPART	NAME
CHEM	Chemistry
CS	Computer Science
EE	Electrical Engineering
ENGL	English Language
MATH	Mathematics
PHYS	Physics
STAT	Statistics

- جدول المواد الدراسية (COURSE_T):

COURSE_	TITLE	UNITS	DEPART
CHEM101	CHEMISTRY (I)	3	CHEM
CHEM102	CHEMISTRY (II)	3	CHEM
CS101	JAVA PROGRAMMING	3	CS
CS102	SOFTWARE ENGINEERING	3	CS
CS103	C/C++ PROGRAMMING	3	CS
CS104	COMPUTER ARCHITECTURE	3	CS
CS105	INTRODUCTION TO DATABASE SYSTEMS	3	CS
EE101	ELECTRIC CIRCUITS	3	EE
EE102	ELECTRONICS (I)	3	EE
EE103	ELECTRONICS (II)	3	EE
EE104	COMMUNICATION NETWORKS	4	EE
ENGL101	ENGLISH GRAMMAR	2	ENGL
ENGL102	ENGLISH WRITING	3	ENGL
ENGL103	TECHNICAL WRITING	3	ENGL
MATH101	INTRODUCTION To MATHEMATICS	3	MATH
MATH102	DIFFERENTIAL EQUATIONS	3	MATH
MATH103	CALCULUS (I)	3	MATH
MATH104	CALCULUS (II)	3	MATH
MATH106	ALGEBRA	4	MATH
MATH107	COMPUTER MATHEMATICS	3	MATH
PHYS101	PHYSICS (I)	3	PHYS
PHYS102	PHYSICS (II)	3	PHYS
STAT101	INTRODUCTION TO STATISTICS	3	STAT
STAT102	ADVANCED STATISTICS	3	STAT

- جدول المواد الدراسية المتطلبة (PREREQUISITE_T):

COURSE_	PREREQU
CHEM102	CHEM101
CS102	MATH101
CS103	CS102
CS105	MATH101
EE102	EE101
EE103	EE102
EE103	MATH101
MATH102	MATH101
MATH103	MATH101
MATH104	MATH103
MATH106	MATH101
MATH107	MATH101
PHYS102	PHYS101
STAT102	STAT101

- جدول أعضاء هيئة التدريس (FUCULTY_T):

FACULTY_	FNAME	LNAME	PHONE_NO	SALARY	DOB	DEPART
200	Khalid	Aloufi	454-2341	35000	22-MAY-63	MATH
220	Fahad	Alhamid	456-7733	25900	07-OCT-70	MATH
310	Saleh	Aleesa	454-8932	30000	13-SEP-66	CS
320	Mohammed	Alhamad	454-5412	44000	13-MAY-65	CS
330	Ghanim	Alghanim	456-2234	44500	12-AUG-69	CS
340	Ibraheem	Alsaleh	454-1234	25000	20-JAN-70	CS
400	Ahmad	Alotaibi	454-4563	33900	17-MAY-71	CHEM
420	Saleh	Alghamdi	454-2233	44600	13-FEB-69	CHEM
500	Yahya	Khorshid	456-2221	36700	12-MAR-65	ENGL
540	Salem	Alhamad	456-3304	40000	11-SEP-72	ENGL
560	Salman	Albassam	454-7865	33800	13-SEP-68	ENGL
600	Turki	Alturki	456-7891	27800	23-JUL-75	STAT
640	Fahad	Alzaid	456-3322	44300	12-MAY-71	STAT
660	Saud	Alkhalifa	454-9856	44900	13-AUG-72	STAT
710	Mahmood	Alsalem	456-3323	31900	19-FEB-73	PHYS
730	Mishal	Almazid	454-2343	29800	17-SEP-75	PHYS
770	Sultan	Aljasir	456-3212	43300	13-MAY-70	PHYS
800	Ali	Albader	456-7812	45300	22-JUN-66	EE
810	Saad	Alzhrani	454-5578	44200	17-OCT-67	EE
850	Ahmad	Alsabti	456-0120	33900	15-APR-73	EE

- جدول القدرات التعليمية لأعضاء هيئة التدريس (QUALIFICATION_T):

COURSE_	FACULTY_	DATE_QUAL
CHEM101	400	02-JAN-91
CHEM102	420	02-JUL-92
CS101	310	05-JUN-95
CS102	320	09-AUG-95
CS103	320	03-AUG-96
CS104	330	02-SEP-97
CS105	340	02-DEC-97
EE101	800	08-JAN-93
EE102	810	12-MAR-94
EE103	850	15-NOV-95
EE104	810	03-JAN-96
ENGL101	500	01-JUL-95
ENGL102	540	02-AUG-94
ENGL103	560	09-SEP-93
MATH101	200	13-NOV-91
MATH102	200	02-JUN-93
MATH103	220	02-JUL-93
MATH104	220	13-AUG-93
MATH106	220	17-OCT-94
MATH107	200	10-JAN-95
PHYS101	710	13-JUL-96
PHYS101	770	11-FEB-96
PHYS102	730	02-JAN-97
STAT101	600	15-AUG-93
STAT101	660	03-APR-95
STAT102	640	02-MAY-94

- جدول المجموعات (أو الشعب) الدراسية (SECTION_T):

COURSE_	SECTION_NO	SEMESTER	YEAR	FACULTY_
CHEM101	1	FALL	2000	400
CHEM101	2	FALL	2000	400
CS101	1	FALL	2000	310
CS101	2	FALL	2000	310
CS102	1	SPRING	2000	320
CS103	1	SPRING	2000	320
CS104	1	FALL	2001	330
CS105	1	SPRING	2001	340
EE101	1	FALL	2001	800
EE102	1	SPRING	2001	810
ENGL101	1	FALL	2000	500
ENGL102	1	SPRING	2000	540
MATH101	1	FALL	2000	200
MATH102	1	SPRING	2000	200
MATH103	1	FALL	2001	220
MATH104	1	SPRING	2001	220
PHYS101	1	FALL	2001	710
PHYS102	1	SPRING	2001	730
STAT101	1	SPRING	2000	600
STAT102	1	SPRING	2001	640

- جدول الطلاب (STUDENT_T):

STUDENT_	FNAME	LNAME	STREET	CITY	ZIP_C	MAJOR
19992020	Saleh	Alhamad	13 Almutanabi Street	Riyadh	11121	CS
19992341	Abdullah	Aloufi	25 Jareer Street	Riyadh	12123	CHEM
19994512	Salem	Algamdi	98 Bin Taimiah Street	Jeddah	34565	PHYS
20001111	Mishal	Alyousef	13 Alsouk Street	Taif	67156	CS
20001212	Khalid	Alsultan	22 Bin Hamdan Street	Jeddah	34565	MATH
20001213	Mohammed	Abdelaleen	10 Bin Hamdan Street	Jeddah	35787	STAT
20001214	Sami	Aloutaibi	67 Alfadel Street	Damman	26123	ENGL
20001215	Saud	Alganim	24 Alfadel Street	Damman	27145	EE
20011212	Abdulrahman	Abdulsalam	10 Almadinah Street	Skaka	88756	CHEM
20011213	Salman	Alsaleh	15 King Fahad Road	Damman	28898	PHYS
20011214	Khalid	Alomar	91 Alwadi Street	Najran	90987	MATH
20011215	Minwer	Almutairi	87 Alhamra Road	Jizan	92347	STAT
20011216	Turki	Alassaf	25 Prince Abdullah Street	Riyadh	11897	ENGL
20011217	Saleh	Alzaid	25 King Faisal Street	Riyadh	11874	EE
20021111	Ghanim	Alhmoud	56 Altahliah Street	Jeddah	35234	CS
20021212	Sultan	Abdulgader	123 Salman Alfarsi Street	Riyadh	12657	CHEM
20021213	Suliman	Almushari	45 Prince Sultan Street	Najran	90888	PHYS
20021214	Ahmad	Alsaif	13 Khalifa Street	Taif	67898	MATH
20021234	Ahmad	Alshemamri	15 Othman street	Jizan	92534	ENGL
20022345	Mohammed	Alzamil	67 Abubaker Road	Abha	56879	STAT
20023678	Mansour	Alzamil	13 King Abdulaziz Road	Tabouk	78453	EE

- جدول تسجيل الطلبة (ENROLLMENT_T):

```
COURSE_  SECTION_NO SEMESTER         YEAR STUDENT_      GRADE
-------  ---------- ---------- --------- --------  ---------
CHEM101           1 FALL             2000 19992020          4
CHEM101           1 FALL             2000 19992341          3
CHEM101           1 FALL             2000 20001212          4
CHEM101           2 FALL             2000 19994512          3
CHEM101           2 FALL             2000 20001111          1
CS101             1 FALL             2000 19992020          2
CS101             2 FALL             2000 20001111          4
CS102             1 SPRING           2000 19992020          3
CS102             1 SPRING           2000 20001111          4
ENGL101           1 FALL             2000 19992020          3
ENGL101           1 FALL             2000 19992341          4
ENGL101           1 FALL             2000 19994512          4
ENGL101           1 FALL             2000 20001111          4
ENGL102           1 SPRING           2000 19992020          1
ENGL102           1 SPRING           2000 20001111          4
MATH101           1 FALL             2000 19992020          3
MATH101           1 FALL             2000 19992341          2
MATH101           1 FALL             2000 19994512          0
MATH101           1 FALL             2000 20001111          2
MATH102           1 SPRING           2000 19992020          2
MATH102           1 SPRING           2000 20001111          0
STAT101           1 SPRING           2000 19992020          2
STAT101           1 SPRING           2000 20001111          3
```

ملحق رقم (٢): تمارين تطبيقية على لغة الاستفسار البنائية (SQL)

١- ما أرقام أعضاء هيئة التدريس العاملين فى الجامعة؟

الحل:

```
SELECT Faculty_ID
FROM FACULTY_T;
```

النتيجة:

```
FACULTY_
--------
200
220
310
320
330
340
400
420
500
540
560
600
640
660
710
730
770
800
810
850
```

٢- ما هى كل بيانات أعضاء هيئة التدريس العاملين فى الجامعة؟

الحل:

```
SELECT *
FROM FACULTY_T;
```

النتيجة:

FACULTY_	FNAME	LNAME	PHONE_NO	SALARY	DOB	DEPART
200	Khalid	Aloufi	454-2341	35000	22-MAY-63	MATH
220	Fahad	Alhamid	456-7733	25900	07-OCT-70	MATH
310	Saleh	Aleesa	454-8932	30000	13-SEP-66	CS
320	Mohammed	Alhamad	454-5412	44000	13-MAY-65	CS
330	Ghanim	Alghanim	456-2234	44500	12-AUG-69	CS
340	Ibraheem	Alsaleh	454-1234	25000	20-JAN-70	CS
400	Ahmad	Alotaibi	454-4563	33900	17-MAY-71	CHEM
420	Saleh	Alghamdi	454-2233	44600	13-FEB-69	CHEM
500	Yahya	Khorshid	456-2221	36700	12-MAR-65	ENGL
540	Salem	Alhamad	456-3304	40000	11-SEP-72	ENGL
560	Salman	Albassam	454-7865	33800	13-SEP-68	ENGL
600	Turki	Alturki	456-7891	27800	23-JUL-75	STAT
640	Fahad	Alzaid	456-3322	44300	12-MAY-71	STAT
660	Saud	Alkhalifa	454-9856	44900	13-AUG-72	STAT
710	Mahmood	Alsalem	456-3323	31900	19-FEB-73	PHYS
730	Mishal	Almazid	454-2343	29800	17-SEP-75	PHYS
770	Sultan	Aljasir	456-3212	43300	13-MAY-70	PHYS
800	Ali	Albader	456-7812	45300	22-JUN-66	EE
810	Saad	Alzhrani	454-5578	44200	17-OCT-67	EE
850	Ahmad	Alsabti	456-0120	33900	15-APR-73	EE

٣- ما أرقام المواد الدراسية وأرقام المواد الدراسية المتطلبة لكل منها؟

الحل:

```
SELECT Course_ID, Prerequisite_ID
FROM PREREQUISITE_T;
```

النتيجة:

COURSE_	PREREQU
CHEM102	CHEM101
CS102	MATH101
CS103	CS102
CS105	MATH101
EE102	EE101
EE103	EE102
EE103	MATH101
MATH102	MATH101
MATH103	MATH101
MATH104	MATH103
MATH106	MATH101
MATH107	MATH101
PHYS102	PHYS101
STAT102	STAT101

٤- ما أرقام أعضاء هيئة التدريس وأرقام البرامج المؤهلين لتدريسها؟

الحل:

```
SELECT Faculty_ID, Course_ID
FROM QUALIFICATION_T;
```

النتيجة:

```
FACULTY_ COURSE_
-------- -------
400      CHEM101
420      CHEM102
310      CS101
320      CS102
320      CS103
330      CS104
340      CS105
800      EE101
810      EE102
850      EE103
810      EE104
500      ENGL101
540      ENGL102
560      ENGL103
200      MATH101
200      MATH102
220      MATH103
220      MATH104
220      MATH106
200      MATH107
710      PHYS101
770      PHYS101
730      PHYS102
600      STAT101
660      STAT101
640      STAT102
```

٥- ما بيانات أعضاء هيئة التدريس الذين يعملون فى قسم الحاسب الآلى (CS)؟

الحل:

```
SELECT *
FROM FACULTY_T
WHERE Department_ID = 'CS';
```

النتيجة:

FACULTY_	FNAME	LNAME	PHONE_NO	SALARY	DOB	DEPART
310	Saleh	Aleesa	454-8932	30000	13-SEP-66	CS
320	Mohammed	Alhamad	454-5412	44000	13-MAY-65	CS
330	Ghanim	Alghanim	456-2234	44500	12-AUG-69	CS
340	Ibraheem	Alsaleh	454-1234	25000	20-JAN-70	CS

٦- ما أرقام وأسماء وأرقام هواتف أعضاء هيئة التدريس الذين يعملون فى قسم اللغة الإنجليزية (ENGL)؟

الحل:

```
SELECT Faculty_ID, FNAME, LNAME, Phone_No
FROM FACULTY_T
WHERE Department_ID = 'ENGL';
```

النتيجة:

FACULTY_	FNAME	LNAME	PHONE_NO
500	Yahya	Khorshid	456-2221
540	Salem	Alhamad	456-3304
560	Salman	Albassam	454-7865

٧- ما أسماء الطلبة من مدينة الرياض (Riyadh)؟

الحل:

```
SELECT FName, LName
FROM STUDENT_T
WHERE CITY = 'Riyadh';
```

النتيجة:

FNAME	LNAME
Saleh	Alhamad
Abdullah	Aloufi
Turki	Alassaf
Saleh	Alzaid
Sultan	Abdulgader

٨- ما أرقام وأسماء وتواريخ ميلاد (DOB) أعضاء هيئة التدريس الذين تواريخ ميلادهم أكبر من ('20-JAN-1970')؟

الحل:

```
SELECT Faculty_ID, FNAME, LNAME, DOB
FROM FACULTY_T
WHERE DOB > '20-JAN-1970'
```

النتيجة:

```
FACULTY_ FNAME        LNAME        DOB
-------- ------------ ------------ ---------
220      Fahad        Alhamid      07-OCT-70
400      Ahmad        Alotaibi     17-MAY-71
540      Salem        Alhamad      11-SEP-72
600      Turki        Alturki      23-JUL-75
640      Fahad        Alzaid       12-MAY-71
660      Saud         Alkhalifa    13-AUG-72
710      Mahmood      Alsalem      19-FEB-73
730      Mishal       Almazid      17-SEP-75
770      Sultan       Aljasir      13-MAY-70
850      Ahmad        Alsabti      15-APR-73
```

٩- ما أرقام وأسماء أعضاء هيئة التدريس المؤهلين لتدريس المادة الدراسية (STAT101)؟

الحل:

```
SELECT FACULTY_T.Faculty_ID, FNAME, LNAME
FROM FACULTY_T, QUALIFICATION_T
WHERE FACULTY_T.Faculty_ID = QUALIFICATION_T.Faculty_ID
AND Course_ID = 'STAT101';
```

النتيجة:

```
FACULTY_ FNAME        LNAME
-------- ------------ ------------
600      Turki        Alturki
660      Saud         Alkhalifa
```

١٠- ما أرقام وأسماء وتقديرات الطلبة الذين درسوا المادة الدراسية (STAT101)؟

الحل:

```
SELECT STUDENT_T.Student_ID, FName, LName, Grade
FROM STUDENT_T, ENROLLMENT_T
WHERE STUDENT_T.Student_ID = ENROLLMENT_T.Student_ID
AND Course_ID = 'STAT101';
```

النتيجة:

```
STUDENT_ FNAME        LNAME            GRADE
-------- ------------ ------------ ---------
19992020 Saleh        Alhamad              2
20001111 Mishal       Alyousef             3
```

١١- مــا أرقــام المواد الدراســية التى لم تنفذ فــى فصل الخريــف (FALL) من عام (2000)؟

الحل:

```
SELECT Course_ID
FROM COURSE_T
WHERE Course_ID NOT IN
(SELECT Course_ID
FROM SECTION_T
WHERE Semester = 'FALL' AND Year = 2000);
```

النتيجة:

```
COURSE_
-------
CHEM102
CS102
CS103
CS104
CS105
EE101
EE102
EE103
EE104
ENGL102
ENGL103
MATH102
MATH103
MATH104
MATH106
MATH107
PHYS101
PHYS102
STAT101
STAT102
```

١٢- ما أسماء أعضاء هيئة التدريس وأسماء الأقسام التى يتبعونها مرتبة تصاعدياً حسب اختصارات أسماء الأقسام التى يتبعونها؟

الحل:

```
SELECT FName, LName, Name
FROM FACULTY_T, DEPARTMENT_T
WHERE FACULTY_T.Department_ID = DEPARTMENT_T.Department_ID
ORDER BY FACULTY_T.Department_ID
```

النتيجة:

```
FNAME        LNAME        NAME
------------ ------------ ------------------------
Ahmad        Alotaibi     Chemistry
Saleh        Alghamdi     Chemistry
Saleh        Aleesa       Computer Science
Mohammed     Alhamad      Computer Science
Ghanim       Alghanim     Computer Science
Ibraheem     Alsaleh      Computer Science
Ali          Albader      Electrical Engineering
Saad         Alzhrani     Electrical Engineering
Ahmad        Alsabti      Electrical Engineering
Yahya        Khorshid     English Language
Salem        Alhamad      English Language
Salman       Albassam     English Language
Khalid       Aloufi       Mathematics
Fahad        Alhamid      Mathematics
Mahmood      Alsalem      Physics
Mishal       Almazid      Physics
Sultan       Aljasir      Physics
Turki        Alturki      Statistics
Fahad        Alzaid       Statistics
Saud         Alkhalifa    Statistics
```

١٣- ما أسماء أعضاء هيئة التدريس وتواريخ ميلادهم وأسماء الأقسام التى يتبعونها مرتبة تصاعدياً حسب تواريخ ميلادهم وتصاعدياً داخل أسماء الأقسام التى يتبعونها (كاملة)؟

الحل:

```
SELECT FName, LName, DOB, Name
FROM FACULTY_T, DEPARTMENT_T
WHERE FACULTY_T.Department_ID = DEPARTMENT_T.Department_ID
ORDER BY Name, DOB;
```

النتيجة:

FNAME	LNAME	DOB	NAME
Saleh	Alghamdi	13-FEB-69	Chemistry
Ahmad	Alotaibi	17-MAY-71	Chemistry
Mohammed	Alhamad	13-MAY-65	Computer Science
Saleh	Aleesa	13-SEP-66	Computer Science
Ghanim	Alghanim	12-AUG-69	Computer Science
Ibraheem	Alsaleh	20-JAN-70	Computer Science
Ali	Albader	22-JUN-66	Electrical Engineering
Saad	Alzhrani	17-OCT-67	Electrical Engineering
Ahmad	Alsabti	15-APR-73	Electrical Engineering
Yahya	Khorshid	12-MAR-65	English Language
Salman	Albassam	13-SEP-68	English Language
Salem	Alhamad	11-SEP-72	English Language
Khalid	Aloufi	22-MAY-63	Mathematics
Fahad	Alhamid	07-OCT-70	Mathematics
Sultan	Aljasir	13-MAY-70	Physics
Mahmood	Alsalem	19-FEB-73	Physics
Mishal	Almazid	17-SEP-75	Physics
Fahad	Alzaid	12-MAY-71	Statistics
Saud	Alkhalifa	13-AUG-72	Statistics
Turki	Alturki	23-JUL-75	Statistics

١٤- ما أسماء أعضاء هيئة التدريس وتواريخ ميلادهم وأسماء الأقسام التى يتبعونها مرتبة تنازلياً حسب تواريخ ميلادهم وتنازلياً داخل أسماء الأقسام التى يتبعونها (كاملة)؟

الحل:

```
SELECT FName, LName, DOB, Name
FROM FACULTY_T, DEPARTMENT_T
WHERE FACULTY_T. Department_ID = DEPARTMENT_T.Department_ID
ORDER BY Name DESC, DOB DESC;
```

النتيجة:

FNAME	LNAME	DOB	NAME
Turki	Alturki	23-JUL-75	Statistics
Saud	Alkhalifa	13-AUG-72	Statistics
Fahad	Alzaid	12-MAY-71	Statistics
Mishal	Almazid	17-SEP-75	Physics
Mahmood	Alsalem	19-FEB-73	Physics
Sultan	Aljasir	13-MAY-70	Physics
Fahad	Alhamid	07-OCT-70	Mathematics
Khalid	Aloufi	22-MAY-63	Mathematics
Salem	Alhamad	11-SEP-72	English Language
Salman	Albassam	13-SEP-68	English Language
Yahya	Khorshid	12-MAR-65	English Language
Ahmad	Alsabti	15-APR-73	Electrical Engineering
Saad	Alzhrani	17-OCT-67	Electrical Engineering
Ali	Albader	22-JUN-66	Electrical Engineering
Ibraheem	Alsaleh	20-JAN-70	Computer Science
Ghanim	Alghanim	12-AUG-69	Computer Science
Saleh	Aleesa	13-SEP-66	Computer Science
Mohammed	Alhamad	13-MAY-65	Computer Science
Ahmad	Alotaibi	17-MAY-71	Chemistry
Saleh	Alghamdi	13-FEB-69	Chemistry

١٥- ما أرقام الطلبة ومعدلاتهم التراكمية؟

الحل:

```
SELECT S.STUDENT_ID, SUM(Grade * Units) / SUM(Units) GPA
FROM STUDENT_T S, ENROLLMENT_T E, COURSE_T  C
WHERE S.Student_ID = E.Student_ID AND E.COURSE_ID = C.COURSE_ID
GROUP BY S.Student_ID;
```

النتيجة:

STUDENT_	GPA
19992020	2.47826087
19992341	2.875
19994512	2.125
20001111	2.69565217
20001212	4

١٦- مــن الطلبة الذين درســوا فى المادة الدراســية (MATH101) أو المادة الدراســية (MATH102) وكان تقديرهم ممتاز (٤,٠٠) أو جيد جداً (٣,٠٠)؟

الحل:

```
SELECT FName, LName
FROM STUDENT_T, ENROLLMENT_T
WHERE STUDENT_T.Student_ID = ENROLLMENT_T.Studetn_ID AND
(Grade = 4 OR Grade = 3) AND
(Course_ID = 'MATH101' OR Course_ID = 'MATH102');
```

النتيجة:

```
FNAME        LNAME
------------ ------------
Saleh        Alhamad
```

١٧- مــا أرقام وأســماء أعضاء هيئــة التدريس الذين تحتوى أســماؤهم الأولى على الحرفين (SA) فى أى موقع بالاسم سواء كانت الحروف بالحجم الصغير أو الكبير (Capital or small letters)؟ رتب الأســماء تصاعدياً حســب الاسم الأول وتصاعدياً داخل اسم العائلة؟

الحل:

```
SELECT Faculty_ID, FName, LName
FROM FACULTY_T
WHERE FName LIKE '%Sa%' OR FName LIKE '%SA%' OR
FName LIKE '%sa%' OR FName LIKE '%sA%'
ORDER BY LName, FName;
```

النتيجة:

```
FACULTY_ FNAME        LNAME
-------- ------------ ----------
560      Salman       Albassam
310      Saleh        Aleesa
420      Saleh        Alghamdi
540      Salem        Alhamad
660      Saud         Alkhalifa
810      Saad         Alzhrani
```

١٨- ما أســماء وأرقام هواتف وأســماء أقســام أعضاء هيئة التدريــس الذين تبدأ أســماؤهم الأولى بالحرف (M) أو تنتهى بالحرف (D)؟ رتب الأســماء تصاعدياً حسب الاسم الأول وتصاعدياً داخل اسم العائلة؟

الحل:

```
SELECT LName, FName, Name
FROM FACULTY_T, DEPARTMENT_T
WHERE FACULTY_T.Department_ID = DEPARTMENT_T.Department_ID AND
(FName LIKE 'M%' OR LIKE 'm%' OR FName LIKE '%D %' OR
FName LIKE '%d %' OR FName LIKE '%d%' OR FName LIKE '%D'
ORDER BY LName, FName;
```

النتيجة:

LNAME	FNAME	NAME
Alhamad	Mohammed	Computer Science
Alhamid	Fahad	Mathematics
Alkhalifa	Saud	Statistics
Almazid	Mishal	Physics
Alotaibi	Ahmad	Chemistry
Aloufi	Khalid	Mathematics
Alsabti	Ahmad	Electrical Engineering
Alsalem	Mahmood	Physics
Alzaid	Fahad	Statistics
Alzhrani	Saad	Electrical Engineering

١٩- ما أسماء أعضاء هيئة التدريس التابعين لقسم الحاسب الآلى بعد زيادة مرتباتهم بمقدار (١٠٪)؟

الحل:

```
SELECT FName, LName, (Salary * 1.1)
FROM FACULTY_T
WHERE FACULTY_T.Department_ID = 'CS';
```

النتيجة:

FNAME	LNAME	(SALARY*1.1)
Saleh	Aleesa	33000
Mohammed	Alhamad	48400
Ghanim	Alghanim	48950
Ibraheem	Alsaleh	27500

٢٠- ما أسماء وتواريخ ميلاد ومرتبات أعضاء هيئة التدريس بعد رفعها بمقدار (١٥٪) للذين ولدوا قبل (01-JUL-1965) ومرتباتهم أقل من أو تساوى (٤٠,٠٠٠)؟ رتب النتيجة تصاعدياً حسب تواريخ الميلاد.

الحل:

```
SELECT FName, LName, DOB, (Salary * 1.15)
FROM FACULTY_T
WHERE FACULTY_T.DOB < '01-JUL-1965' AND Salary < = 40000
ORDER BY DOB;
```

النتيجة:

FNAME	LNAME	DOB	(SALARY*1.15)
Khalid	Aloufi	22-MAY-63	40250
Yahya	Khorshid	12-MAR-65	42205

٢١- ما أسماء وتواريخ ميلاد ومرتبات أعضاء هيئة التدريس بعد رفعها بمقدار (١٥٪) للذين ولدوا بعد (01-JUL-1970) ومرتباتهم أكبر من (٤٠,٠٠٠)؟ رتب النتيجة تصاعدياً حسب المرتبات بعد الرفع.

الحل:

```
SELECT FName, LName, DOB, (Salary * 1.15)
FROM FACULTY_T
WHERE FACULTY_T.DOB > '01-JUL-1970' AND Salary > 40000
ORDER BY 4;
```

النتيجة:

```
FNAME        LNAME        DOB       (SALARY*1.15)
------------ ------------ --------- -------------
Fahad        Alzaid       12-MAY-71         50945
Saud         Alkhalifa    13-AUG-72         51635
```

٢٢- ما أسماء أعضاء هيئة التدريس الذين تم تأهيلهم لتدريس إحدى المواد الدراسية بعد تاريخ (1-MAR-1996) ؟

الحل:

```
SELECT FName, LName
FROM FACULTY_T, QUALIFICATION_T
WHERE FACULTY_T.Faculty_ID = QUALIFICATION_T.Faculty_ID AND
Date_Qualified > = '1-MAR-1996';
```

النتيجة:

```
FNAME        LNAME
------------ ------------
Mohammed     Alhamad
Ghanim       Alghanim
Ibraheem     Alsaleh
Mahmood      Alsalem
Mishal       Almazid
```

٢٣- مـــا أكبر راتب يتقاضاه أعضـــاء هيئة التدريس فى الجامعة؟ أظهر النتيجة تحت مسمى (MAXIMUM_SALARY)؟

الحل:

```
SELECT MAX(Salary) MAXIMUM_SALARY
FROM FACULTY_T;
```

النتيجة:

```
MAXIMUM_SALARY
--------------
         45300
```

٢٤- ما أكبر راتب يتقاضاه أعضاء هيئة التدريس فى قسم الحاسب الآلى؟ أظهر النتيجة تحت مسمى (MAXIMUM_SALARY).

الحل:

```
SELECT MAX(Salary) MAXIMUM_SALARY
FROM FACULTY_T
WHERE Department_ID = 'CS';
```

النتيجة:

```
MAXIMUM_SALARY
--------------
         44500
```

٢٥- ما عدد أعضاء هيئة التدريس العاملين فى قسم الحاسب الآلى (CS)؟ وما مجموع ومتوسط مرتباتهم وأكبر وأصغر مرتب؟

الحل:

```
SELECT COUNT(*), SUM(Salary), AVG(Salary), MAX(Salary), MIN(Salary)
FROM FACULTY_T
WHERE Department_ID = 'CS';
```

النتيجة:

```
 COUNT(*) SUM(SALARY) AVG(SALARY) MAX(SALARY) MIN(SALARY)
--------- ----------- ----------- ----------- -----------
        4      143500       35875       44500       25000
```

٢٦- ما مجموع ومتوسط رواتب أعضاء هيئة التدريس الذين يعملون فى قسم الرياضيات (MATH)؟ أظهر النتيجة تحت مسمى (SUM_SALARY) ومسمى (AVG_SALARY).

الحل:

```
SELECT SUM(Salary) SUM_SALARY, AVG(Salary) AVG_SALARY
FROM FACULTY_T
WHERE Department_ID = 'MATH';
```

النتيجة:

```
SUM_SALARY AVG_SALARY
---------- ----------
     60900      30450
```

٢٧- ما عدد أعضاء هيئة التدريس الذين لا يعملون في قسم الحاسب الآلي وتتراوح مرتباتهم ما بين (٣٠,٠٠٠) و (٤٠,٠٠٠)، وما مجموع مرتباتهم؟ أظهر عدد أعضاء هيئة التدريس تحت مسمى (No_of_Faculty).

الحل:

```
SELECT COUNT(*) No_fo_Faculty, SUM(Salary)
FROM FACULTY_T
WHERE Department_ID <> 'CS' AND Salary > = 3000 AND Salary < = 40000;
```

النتيجة:

```
NO_OF_FACULTY SUM(SALARY)
------------- -----------
            7      245200
```

٢٨- ما مجموع ومتوسط مرتبات أعضاء هيئة التدريس الذين يعملون في قسم الحاسب الآلي قبل وبعد رفعها بمقدار (١٠٪)؟ أظهر المجموع قبل الزيادة تحت مسمى (Sum_Before_Increase) والمجموع بعد الزيادة تحت مسمى (Sum_After_Increase)، والمتوسط قبل الزيادة تحت مسمى (Avg_Before_Increase)، والمتوسط بعد الزيادة تحت مسمى (Avg_After_Increase).

الحل:

```
SELECT SUM(Salary) Sum_Before_Increase, SUM(Salary * 1.1) Sum_After_Increae,
AVG(Salary) Avg_Before_Increase, AVG(Salary * 1.1) Avg_After_Increase
FROM FACULTY_T
WHERE Department_ID = 'CS';
```

النتيجة:

```
SUM_BEFORE_INCREASE SUM_AFTER_INCREASE AVG_BEFORE_INCREASE AVG_AFTER_INCREASE
------------------- ------------------ ------------------- ------------------
             143500             157850               35875            39462.5
```

٢٩- لكل قسم من أقسام الجامعة، ما أكبر وأصغر ومتوسط مرتبات أعضاء هيئة التدريس فى القسم؟

الحل:

```
SELECT Department_ID, MAX(Salary), MIN(Salary), AVG(Salary)
FROM FACULTY_T
GROUP BY Department_ID;
```

النتيجة:

```
DEPART MAX(SALARY) MIN(SALARY) AVG(SALARY)
------ ----------- ----------- -----------
CHEM         44600       33900       39250
CS           44500       25000       35875
EE           45300       33900   41133.333
ENGL         40000       33800   36833.333
MATH         35000       25900       30450
PHYS         43300       29800       35000
STAT         44900       27800       39000
```

٣٠- ما عدد الطلبة المسجلين فى كل مادة دراسية على حدة؟

الحل:

```
SELECT Course_ID, COUNT(*) NO_OF_STUDENTS
FROM ENROLLMENT_T
GROUP BY Course_ID;
```

النتيجة:

```
COURSE_ NO_OF_STUDENTS
------- --------------
CHEM101              5
CS101                2
CS102                2
ENGL101              4
ENGL102              2
MATH101              4
MATH102              2
STAT101              2
```

٣١- ما عدد الطلبة المسجلين فى كل مادة دراسية من المواد التى ينفذها قسم الحاسب الآلى (CS)؟

الحل:

```
SELECT Course_ID, COUNT(*) NO_OF_STUDENTS
FROM ENROLLMENT_T
WHERE Course_ID LIKE 'CS%'
GROUP BY Course_ID;
```

أو

```
SELECT Course_ID, COUNT(*) NO_OF_STUDENTS
FROM ENROLLMENT_T
WHERE Course_ID IN
(SELECT Course_ID
FROM COURSE_T
WHERE Department_ID = 'CS')
GROUP BY Course_ID;
```

النتيجة:

```
COURSE_ NO_OF_STUDENTS
------- --------------
CS101                2
CS102                2
```

٣٢- ما رقم كل مادة دراسية سجل فيها أربعة طلبة؟ وما متوسط التقديرات فيها؟

الحل:

```
SELECT Course_ID, AVG(Grade) AVERAGE_GRADE
FROM ENROLLMENT_T EN1
WHERE (SELECT  COUNT(Student_ID)
FROM ENROLLMENT_T
WHERE Course_ID = EN1.Course_ID) = 4
GROUP BY Course_ID;
```

النتيجة:

```
COURSE_ AVERAGE_GRADE
------- -------------
ENGL101          3.75
MATH101          1.75
```

٣٣- ما مجموع مرتبات أعضاء هيئة التدريس بعد رفعها بنسبة (١٥٪) فى كل قسم من أقسام الجامعة عدا قسم الرياضيات (MATH)؟ رتب النتيجة تصاعدياً حسب المجموع.

الحل:

```
SELECT Department_ID, SUM(Salaty * 1.15)
FROM FACULTY_T
WHERE Department_ID <> 'MATH'
GROUP BY Department_ID
ORDER BY 2;
```

النتيجة:

```
DEPART SUM(SALARY*1.15)
------ ----------------
CHEM              90275
PHYS             120750
ENGL             127075
STAT             134550
EE               141910
CS               165025
```

٣٤- ما أرقام المواد الدراسية التى نفذت من خلال أكثر من مجموعة (أو شعبة)؟ رتب النتيجة تصاعدياً حسب رمز المادة الدراسية.

الحل:

```
SELECT Course_ID
FROM SECTION_T
GROUP BY Course_ID
HAVING COUNT(Course_ID) > 1;
```

النتيجة:

```
COURSE_
-------
CHEM101
CS101
```

٣٥- ما أســماء أعضاء هيئة التدريس وأرقام المواد الدراســية التى درسوا المجموعة الأولــى منهــا (فى أى برنامــج كان) خلال الفصل الدراســى (SPRING) من عام (2000)؟

الحل:

```
SELECT FName, LName, Course_ID
FROM FACULTY_T, SECTION_T
WHERE FACULTY_T.Faculty_ID =
SECTION_T. Faculty_ID AND Section_No = 1 AND
Semester = 'SPRING' AND Year = 2000;
```

النتيجة:

```
FNAME        LNAME        COURSE_
------------ ------------ -------
Mohammed     Alhamad      CS102
Mohammed     Alhamad      CS103
Salem        Alhamad      ENGL102
Khalid       Aloufi       MATH102
Turki        Alturki      STAT101
```

٣٦- ما أرقام الطلبة وتقديراتهم وأرقام المواد الدراســية التى درســوها وحصلوا على تقديرات ٣ أو ٤؟ رتب النتيجة تنازلياً حسب التقديرات مرتبة بشكل تنازلى داخل أرقام المواد الدراسية التى درسوها.

الحل:

```
SELECT Student_ID, Course_ID, Grade
FROM ENROLLMENT_T
WHERE Grade = 3 OR Grade = 4
ORDER BY Course_ID DESC, Grade  DESC;
```

النتيجة:

```
STUDENT_  COURSE_      GRADE
--------  -------  ---------
20001111  STAT101          3
19992020  MATH101          3
20001111  ENGL102          4
19992341  ENGL101          4
19994512  ENGL101          4
20001111  ENGL101          4
19992020  ENGL101          3
20001111  CS102            4
19992020  CS102            3
20001111  CS101            4
19992020  CHEM101          4
20001212  CHEM101          4
19992341  CHEM101          3
19994512  CHEM101          3
```

٣٧- ما رقم المادة الدراسـية ومتوسط التقديرات وعدد الطلاب لكل مادة سجل فيها أكثر من طالبين؟

الحل:

```
SELECT Course_ID, AVG(Grade) AVERGE_GRADE,
COUNT(Student_ID) NO_OF_STUDENTS
FROM ENROLLMENT_T
GROUP BY Course_ID
HAVING  COUNT(Course ID) > 2;
```

أو

```
SELECT Course_ID, AVG(Grade) AVERGE_GRADE,
COUNT(Student_ID) NO_OF_STUDENTS
FROM ENROLLMENT_T EN1
WHERE (SELECT COUNT(Student_ID)
FROM ENROLLMENT_T
WHERE Course_ID = EN1.Course_ID) > 2
GROUP BY Course_ID;
```

النتيجة:

```
COURSE_  AVERAGE_GRADE  NO_OF_STUDENTS
-------  -------------  --------------
CHEM101              3               5
ENGL101           3.75               4
MATH101           1.75               4
```

٣٨- اســـتخدم تعليمة إنشاء جدول (CREATE TABLE) لنسخ جدول الأقسام الدراسية (DEPARTMENT_T) بمسمى (MY_DEPARTMENT_T).

الحل:

```
CREATE TABLE MY_DEPARTMENT_T AS
SELECT *
FROM DEPARTMENT_T;
```

٣٩- استخدم تعليمة (ALTER) لإضافة الحقل (أو العمود) (Location) للجدول الجديد الذى قمت بإنشائه بمســـمى (MY_DEPARTMENT_T) بحيث تكون بياناته نصية (Text) وطولها عشرة خانات.

الحل:

```
ALTER TABLE MY_DEPARTMENT_T ADD
Location CHAR(10);
```

٤٠- استخدم تعليمة (UPDATE) لإضافة بيانات العمود (LOCATION) حسب التالى:

اسم القسم (Department Name)	الموقع (Location)
قسم الحاسب الآلى (CS)	الرياض (RIYADH)
قسم الرياضيات (MATH)	الرياض (RIYADH)
قسم الإحصاء (STAT)	جدة (JEDDAH)
قسم الهندسة الكهربائية (EE)	الدمام (DAMMAM)
قسم الكيمياء (CHEM)	الجوف (JOUF)
قسم اللغة الإنجليزية (ENGL)	جازان (JAZAN)
قسم الفيزياء (PHYS)	جدة (JEDDAH)

الحل:

تحديد بيانات الأقسام الدراسية فى الجدول الجديد

```
UPDATE MY_DEPARTMENT_T SET Location = 'RIYADH' WHERE Department_ID = 'CS';
UPDATE MY_DEPARTMENT_T SET Location = 'RIYADH' WHERE Department_ID = 'MATH';
UPDATE MY_DEPARTMENT_T SET Location = 'JEDDAH' WHERE Department_ID = 'STAT';
UPDATE MY_DEPARTMENT_T SET Location = 'DAMMAM' WHERE Department_ID = 'EE';
UPDATE MY_DEPARTMENT_T SET Location = 'JOUF' WHERE Department_ID = 'CHEM';
UPDATE MY_DEPARTMENT_T SET Location = 'JAZAN' WHERE Department_ID = 'ENGL';
UPDATE MY_DEPARTMENT_T SET Location = 'JEDDAH' WHERE Department_ID = 'PHYS';
```

٤١- أنشئ منظوراً (View) بمسمى (COURSE_LIST_V) يدمج الحقول (Course_ID) و (Title) من جدول المواد الدراسية (COURSE_T) مع الحقل (Name) والحقل (Location) من الجدول الجديد الذى قمت بإنشائه (MY_DEPARTMENT_T).

الحل:

```
CREATE VIEW COURSE_LIST_V AS
SELECT Course_ID, Title, Name, Location
FROM  COURSE_T, MY_DEPARTMENT_T
WHERE COURSE_T.Department_ID = MY_DEPARTMENT_T. Department_ID;
```

النتيجة:

```
COURSE_ TITLE                                NAME                           LOCATION
------- ------------------------------------ ------------------------------ ---------
CHEM101 CHEMISTRY (I)                        Chemistry                      JOUF
CHEM102 CHEMISTRY (II)                       Chemistry                      JOUF
CS101   JAVA PROGRAMMING                     Computer Science               RIYADH
CS102   SOFTWARE ENGINEERING                 Computer Science               RIYADH
CS103   C/C++ PROGRAMMING                    Computer Science               RIYADH
CS104   COMPUTER ARCHITECTURE                Computer Science               RIYADH
CS105   INTRODUCTION TO DATABASE SYSTEMS     Computer Science               RIYADH
EE101   ELECTRIC CIRCUITS                    Electrical Engineering         DAMMAM
EE102   ELECTRONICS (I)                      Electrical Engineering         DAMMAM
EE103   ELECTRONICS (II)                     Electrical Engineering         DAMMAM
EE104   COMMUNICATION NETWORKS               Electrical Engineering         DAMMAM
ENGL101 ENGLISH GRAMMAR                      English Language               JAZAN
ENGL102 ENGLISH WRITING                      English Language               JAZAN
ENGL103 TECHNICAL WRITING                    English Language               JAZAN
MATH101 INTRODUCTION To MATHEMATICS          Mathematics                    RIYADH
MATH102 DIFFERENTIAL EQUATIONS               Mathematics                    RIYADH
MATH103 CALCULUS (I)                         Mathematics                    RIYADH
MATH104 CALCULUS (II)                        Mathematics                    RIYADH
MATH106 ALGEBRA                              Mathematics                    RIYADH
MATH107 COMPUTER MATHEMATICS                 Mathematics                    RIYADH
PHYS101 PHYSICS (I)                          Physics                        JEDDAH
PHYS102 PHYSICS (II)                         Physics                        JEDDAH
STAT101 INTRODUCTION TO STATISTICS           Statistics                     JEDDAH
STAT102 ADVANCED STATISTICS                  Statistics                     JEDDAH
```

٤٢- مـــا أرقام المواد الدراســـية (Course_ID) وأســـماؤها (Title) التى تنفذ من خلال أقسام علمية مواقعها (Location) فى مدينة الرياض (RIYADH) أو الجوف (JOUF) وذلك حسب ورودها فى المنظور (COURSE_LIST_V)؟

الحل:

```
SELECT Course_ID, Title
FROM COURSE_LIST_V
WHERE Location = 'RIYADH' OR Location = 'JOUF';
```

النتيجة:

```
COURSE_ TITLE
------- ----------------------------------
CHEM101 CHEMISTRY (I)
CHEM102 CHEMISTRY (II)
CS101   JAVA PROGRAMMING
CS102   SOFTWARE ENGINEERING
CS103   C/C++ PROGRAMMING
CS104   COMPUTER ARCHITECTURE
CS105   INTRODUCTION TO DATABASE SYSTEMS
MATH101 INTRODUCTION To MATHEMATICS
MATH102 DIFFERENTIAL EQUATIONS
MATH103 CALCULUS (I)
MATH104 CALCULUS (II)
MATH106 ALGEBRA
MATH107 COMPUTER MATHEMATICS
```

٤٣- أنشـــئ فهرســـاً للجدول الجديد (MY_DEPARTMENT_T) على مواقع الأقســـام الدراسية (LOCATION) باسم (DEPARTMENT_LOCATION_IDX).

الحل:

```
CREATE INDEX DEPARTMENT_LOCATION_IDX ON MY_DEPARTMENT_T (Location);
```



المؤلف فى سطور

د. يوسف بن جاسم بن محمد الهميلي

المؤهل العلمى:

- الدكتوراه فى هندسة الحاسب الآلى من جامعة بتسبرغ، الولايات المتحدة الأمريكية، عام ١٤١٨هـ.

الوظيفة الحالية:

- أستاذ هندسة الحاسب الآلى المشارك - معهد الإدارة العامة بالرياض.

الأنشطة العملية:

- عمل أستاذاً زائراً فى جامعة واترلوو الكندية (Waterloo, CANADA) (العام الدراسى ١٤٢٦ - ١٤٢٧هـ).
- عمل عضواً فى الأمانة العامة للجنة الوزارية للتنظيم الإدارى (١٤٢٣-١٤٢٤هـ).
- عمل مديراً لبرامج الحاسب الآلى والمعلومات فى معهد الإدارة العامة (١٤١٨-١٤٢٣هـ).
- عمل مستشـــاراً لعدد من الجهات الحكومية فى المملكة العربية الســـعودية، ومنها: ديوان سمو ولى العهد، وهيئة الرقابة والتحقيق، والرئاسة العامة لتعليم البنات (قبل دمجها فى وزارة التربية والتعليم).
- عمل عضواً فى العديد من اللجان الفنية والعلمية بمعهد الإدارة العامة، من ضمنها المجلـــس العلمى، واللجنة الدائمة للمعلومات والتقنية. كما ترأس العديد من اللجان وفرق العمل، من ضمنها فريق إعداد الخطة الإســـتراتيجية للمعلومات والتعاملات الإلكترونية.
- شارك ضمن اللجان المنظمة لعدد من المؤتمرات الدولية، ومن تلك المؤتمرات المؤتمر الدولى الأول والثانى والثالث والخامس فى إعادة استخدام المعلومات وتكاملها "IEEE International Conference on Information Reuse and Integration (IRI)" والمؤتمر الدولى

الأول للحاسبات، والاتصالات، ومعالجة الإشارات "1st IEEE International Conference on Computers, Communications, and Signal Processing"

- قام بتدريس العديد من المواد العلمية المتخصصة فى مجال الحاسب الآلى على مستوى الدبلوم (بعد المرحلة الثانوية) والبكالوريوس والدراسات العليا داخل المملكة العربية السعودية وخارجها.

الأنشطة العلمية:

- قام بتحكيم ومراجعة عدد من المؤلفات العلمية، من ذلك كتب مرجعية ومقالات مقدمة لدوريات علمية ومؤتمرات دولية.

- شارك فى تأليف الموسوعة العالمية لنظم قواعد البيانات (Encyclopedia of Database Systems) من إصدارات دار سبرنغر - فرلاغ (Springer-Verlag) الألمانية للطباعة والنشر، كما أن له أكثر من عشرين بحثاً ومقالة علمية منشورة، من أحدثها ما يلى:

دوريات علمية:

- **Incompatibility Dimensions and Integration of Atomic Commit Protocols,** Yousef J. Al-Houmaily, *Internationl Arab Journal of Informaton Technology*, Vol. 5, No. 4, October 2008.

- **An Atomic Commit Protocol for Gigabit-Networked Distributed Databeses,** Yousef J. Al-Houmaily and Panos K. Chrysanthis, *Journal of Systems Architecture*, The EUROMICRO Journal, Vol. 46, No. 9, June 2000, pp. 809-833.

فصول فى كتب:

- **Recovery and Performance of Atomic Commit Processing in Distributed Database Systems,** Panos K. Chrysanthis, George Samaras and Yousef J. Al-Houmaily. In Recovery Mechanisms in Database Systems, *V. Kumar and M. Hsu, eds.*, pp. 370-416, *Printice Hall, 1998.*

ندوات ومؤتمرات:

- **On Interoperating Incompatible Atomic Commit Protocols in Distributed Databases,** Yousef J. Al-Houmaily, *Proceedings of the 1st IEEE International Conference on Computers, Communications, and Signal Processing*, pp. 149-156, Kuala Lumpur, Malaysia, November 2005.

- **ML-1-2PC: An Adaptive Multi-Level Atomic Commit Protocol,** Yousef J. Al-Houmaily and Panos K.Chrysantis, *Proceedings of the 8th East-Euoropean Conference on Advances in Databases and Information Systems*, Lecture Notes on Computer Science (LNCS), Vol. 3255, pp. 275-290, Budapest, Hungary, Septemper 2004.

- **1-2PC: The One-Two Phase Atomic Commit Protocol,** Yousef J. Al-Houmaily and Panos K. Chrysanthis, *Proceedings of th 19th ACM Annual Symposium on Applied Computing, Special Track on Database Theory, Technology, and applications*, Nicosia, Cyprus, March 2004.

تم التصميم والإخراج الفنى والطباعة فى
الإدارة العامة للطباعة والنشر بمعهد الإدارة العامة - ١٤٢٩هـ

هذا الكتاب

ينظر فى نظم قواعد البيانات، وهى فى وقتنا الراهن واحدة من أخصب التخصصات العلمية طرقاً من قبل الباحثين وتطبيقاً من قبل المتخصصين فى مجال تطوير النظم المعلوماتية؛ وذلك لكونها الأساس الذى تبنى عليه النظم المعلوماتية الحديثة التى تتوافر فى جميع أنواع المنظمات المتطورة، على اختلاف أحجامها، سواء أكانت تعنى بالتعليم، أو الرعاية الصحية، أو الأعمال البنكية وأسواق المال، أو التجارة الإلكترونية، وذلك على سبيل المثال فحسب. وعلى أهمية هذا المجال تعانى المكتبة العربية نقصاً فيه، وقد شجع هذا القصور على تأليف الكتاب.

ويشتمل الكتاب على الموضوعات الرئيسة لقواعد البيانات، مع ميله إلى الجانب التطبيقى دون تقصير فى عرض الجوانب النظرية التى تستند إليها مفاهيم وتقنيات قواعد البيانات. ويستهدف الكتاب الطلبة الدارسين فى مواد نظم قواعد البيانات من المتخصصين فى مجال الحاسب الآلى سواء فى مرحلة الدراسة الجامعية (البكالوريوس) أو طلبة الدبلوم (فوق الثانوى). كما يستهدف أيضاً مطورى نظم التطبيقات الذين يتعاملون مع نظم قواعد البيانات العلاقية بشكل تطبيقى فى حياتهم اليومية، ليصبح مرجعاً تطبيقياً لهم.

ومن السمات الرئيسة لهذا الكتاب ما يلى:

- تسلسل محتوياته بشكل منطقى يتوافق مع تسلسل مراحل تطوير قواعد البيانات.
- تقسيمه لمنهجية تطوير قواعد البيانات إلى ثلاث مراحل: النمذجة المفاهيمية (Conceptual Modeling)، والتصميم المنطقى (Logical Design)، والتصميم المادى (Physical Design).
- شرحه المستفيض لنموذج كينونة - علاقة (Entity-Relationship Model) الذى يعد من أكثر النماذج المفاهيمية شيوعاً.
- شرحه للمفاهيم الأساسية للنموذج العلاقى (Relational Model) الذى يعد من أكثر النماذج التمثيلية استخداماً فى وقتنا الراهن، ولغاته الرسمية.
- استعراضه لتعليمات لغة الاستفسار البنائية القياسية بشكل تطبيقى على قاعدة بيانات أوراكل وقاعدة بيانات أكسس.
- احتواؤه على حالة تطبيقية لقاعدة بيانات افتراضية بهدف إيضاح المفاهيم بشكل تطبيقى فى فصول الكتاب كافةً.
- تضمنه فصلاً، وهو الفصل التاسع، يشرح بعض المفاهيم المتقدمة لنظم قواعد البيانات.

وإضافة إلى تميز الكتاب بتلك السمات، تتجلى فيه كذلك خبرة المؤلف فى التدريس والبحوث العلمية على المستوى الدولى من خلال سلاسة اللغة المستخدمة والعمق فى العرض والتحليل.

ردمك: ١-١٧٠٠-١٤-٩٩٦٠-٩٧٨

تصميم وإخراج وطباعة الإدارة العامة للطباعة والنشر - معهد الإدارة العامة ١٤٢٩هـ